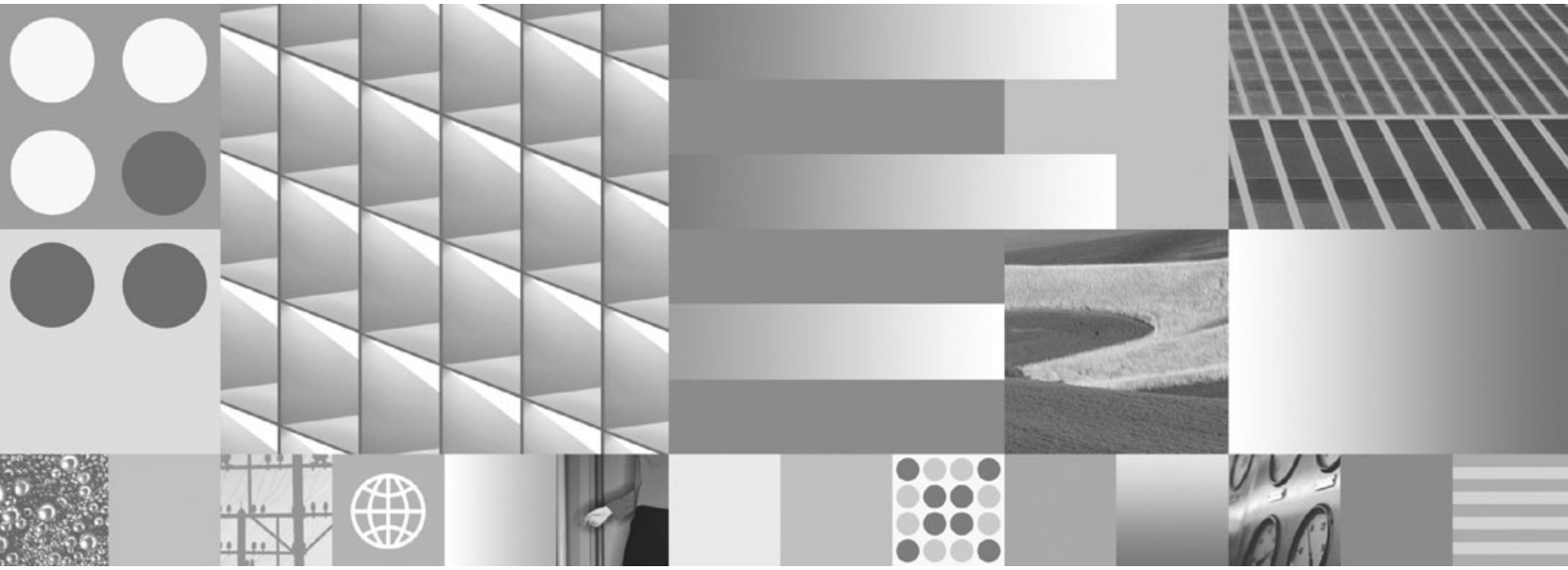
DB2
for Linux, UNIX, and Windows

Version 9 Release 7

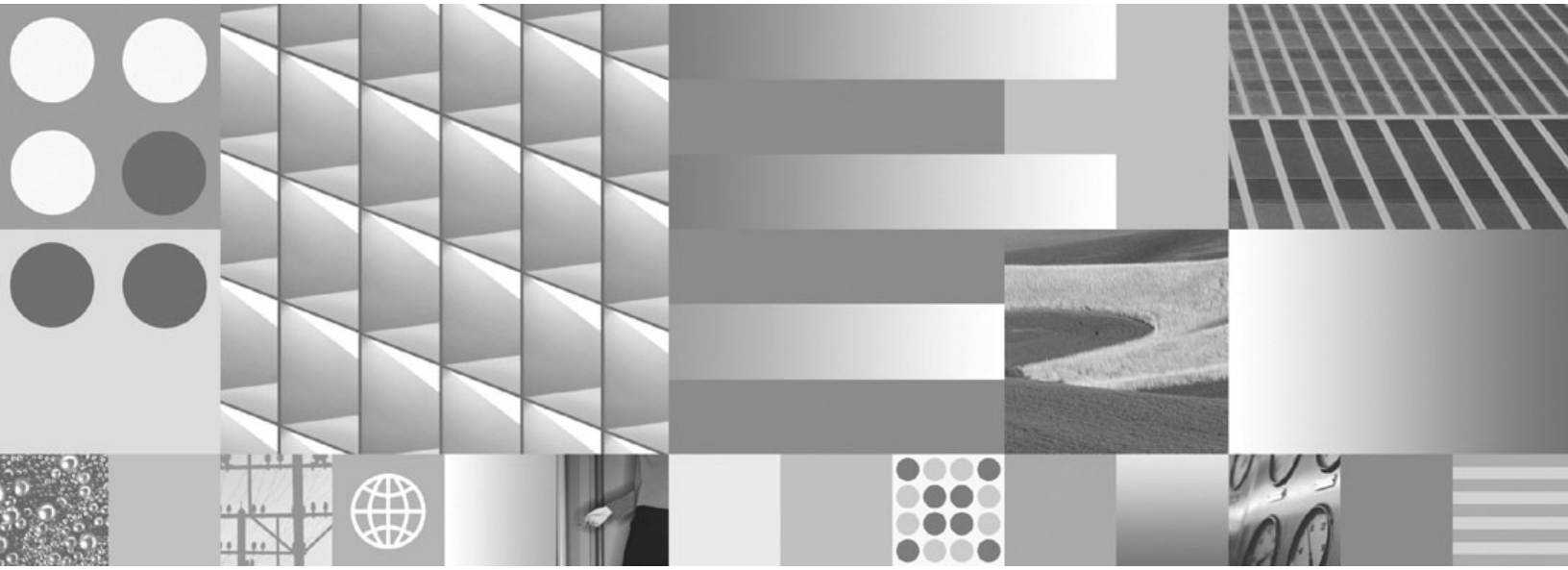


Database Monitoring Guide and Reference
Updated July, 2012

SC27-2458-03

DB2
for Linux, UNIX, and Windows

Version 9 Release 7



Database Monitoring Guide and Reference
Updated July, 2012

SC27-2458-03

Note

Before using this information and the product it supports, read the general information under Appendix B, "Notices," on page 1085.

Edition Notice

This document contains proprietary information of IBM. It is provided under a license agreement and is protected by copyright law. The information contained in this publication does not include any product warranties, and any statements provided in this manual should not be interpreted as such.

You can order IBM publications online or through your local IBM representative.

- To order publications online, go to the IBM Publications Center at www.ibm.com/shop/publications/order
- To find your local IBM representative, go to the IBM Directory of Worldwide Contacts at www.ibm.com/planetwide

To order DB2 publications from DB2 Marketing and Sales in the United States or Canada, call 1-800-IBM-4YOU (426-4968).

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright IBM Corporation 1993, 2012.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About this book xix

Part 1. Monitoring interfaces. 1

Chapter 1. Database monitoring 3

Chapter 2. Monitor table functions overview 5
Monitoring system information using table functions 5
Monitoring activities using table functions 6
Monitoring data objects using table functions . . . 7
Monitoring locking using table functions 8
Monitoring system memory using table functions . . 8
Other monitoring table functions 8

Chapter 3. Interfaces that return monitor data in XML documents 9
Interfaces for viewing XML monitor information as formatted text 14
Viewing metrics monitor elements from XML documents as rows in a table 18

Chapter 4. Event monitors 23
Event monitors that write to an unformatted event table. 25
Unformatted event table column definitions . . 27
db2evmonfmt tool for reading event monitor data 30
Monitoring database locking. 34
Monitoring package cache events 68
Monitoring unit of work events 94
Capturing system metrics using the statistics event monitor 128
Capturing activity monitor elements using the activity event monitor 163
Event monitors that write to tables, files, and pipes 197
Collecting information about database system events 198
Creating an event monitor 200
Event monitor sample output 217

Chapter 5. Reports generated using the MONREPORT module 223
Diagnosing SQL performance slowdowns using MONREPORT module reports. 226
Determining the problem category 227
Identifying the scope of an SQL slowdown . . 230
Reviewing top SQL statements 232
Reviewing all work running on the system . . 233
Collecting statement text and the details of the SQL statements involved 234

Chapter 6. Determining the date a database object was last used 237

Chapter 7. Time-spent monitor elements 239
Hierarchy of time-spent monitor elements 240
Wait times for FCM communications 246
Retrieving and working with time-spent monitor element data 248
Seeing where time is spent across the system 248
Determining where time is spent during SQL statement execution 252

Chapter 8. Snapshot monitor. 255
Access to system monitor data: SYSMON authority 255
Capturing database system snapshots using snapshot administrative views and table functions . 256
Capturing database system snapshot information to a file using the SNAP_WRITE_FILE stored procedure 258
Accessing database system snapshots using snapshot table functions in SQL queries (with file access). 260
Snapshot monitor SQL Administrative Views. . . 261
SQL access to database system snapshots 264
Capturing a database snapshot from the CLP. . . 265
Snapshot monitor CLP commands 266
Capturing a database snapshot from a client application 268
Snapshot monitor API request types. 270
Snapshot monitor sample output 272
Subsection snapshots 274
Global snapshots on partitioned database systems 275
Snapshot monitor self-describing data stream. . . 275
Monitoring with db2top in interactive mode commands 278
.db2toprc configuration file. 280

Chapter 9. Switch-based monitoring concepts 283
System monitor switches 283
Setting system monitor switches from the CLP 285
Setting system monitor switches from a client application 287
System monitor switches self-describing data stream. 288
Database system monitor data organization . . . 289
Counter status and visibility 290
System monitor output: the self-describing data stream. 291
Memory requirements for monitor data 292
Monitoring buffer pool activity 295
Database system monitor interfaces 296

© Copyright IBM Corp. 1993, 2012

Chapter 10. Deprecated monitoring tools 299
Health monitor 299
Monitoring database health. 299
Health indicators 333
Working with the Memory Visualizer 362
Memory Visualizer overview 364
Activity Monitor overview 367
Monitoring scenarios 370
Setting up an activity monitor 373
Progress monitoring of the rollback process . . 373
Using snapshot monitor data to monitor the reorganization of a partitioned table 374
Inactive statement tracking for DEADLOCK WITH DETAILS HISTORY event monitors. . . 382
Introduction to Windows Management Instrumentation (WMI) 383
DB2 database system integration with Windows Management Instrumentation 384
Windows performance monitor introduction . . 385
Indoubt Transaction Manager overview 389

Part 2. Monitor elements 393

Chapter 11. Monitor elements reported in monitor table functions 395

Chapter 12. Request monitor elements 407

Chapter 13. Activity monitor elements 409

Chapter 14. Data object monitor elements 411

Chapter 15. Hierarchy of time-spent monitor elements. 413

Chapter 16. Logical data groups . . . 419
Snapshot monitor interface mappings to logical data groups 419
Snapshot monitor logical data groups and monitor elements 423
Event type mappings to logical data groups . . . 454
Event monitor logical data groups and monitor elements 457
Logical data groups affected by COLLECT ACTIVITY DATA settings 491

Chapter 17. Database system monitor elements 493
acc_curs_blk - Accepted Block Cursor Requests . . 494
act_aborted_total - Total aborted activities monitor element 494
act_completed_total - Total completed activities monitor element 495
act_cpu_time_top – Activity CPU time top monitor element 496
act_exec_time - Activity execution time monitor element 497
act_rejected_total - Total rejected activities monitor element 497
act_remapped_in – Activities remapped in monitor element 498
act_remapped_out – Activities remapped out monitor element 499
act_rows_read_top – Activity rows read top monitor element 499
act_rqsts_total - Total activity requests monitor elements 500
act_total - Activities total monitor element. . . . 501
activate_timestamp - Activate timestamp monitor element 501
active_hash_joins - Active hash joins 501
active_olap_funcs - Active OLAP Functions monitor element 501
active_sorts - Active Sorts 502
activity_collected - Activity collected monitor element 502
activity_id - Activity ID monitor element 502
activity_secondary_id - Activity secondary ID monitor element 503
activity_state - Activity state monitor element . . 504
activity_type - Activity type monitor element. . . 504
activitytotaltime_threshold_id - Activity total time threshold ID monitor element 505
activitytotaltime_threshold_value - Activity total time threshold value monitor element 505
activitytotaltime_threshold_violated - Activity total time threshold violated monitor element 506
adapter_name - Adapter name monitor element 506
address - IP address from which the connection was initiated 506
agent_id - Application handle (agent ID) monitor element 506
agent_id_holding_lock - Agent ID Holding Lock 508
agent_pid - Engine dispatchable unit (EDU) identifier monitor element 509
agent_status - DCS Application Agents 509
agent_sys_cpu_time - System CPU Time used by Agent 509
agent_usr_cpu_time - User CPU Time used by Agent 510
agent_wait_time - Agent wait time monitor element 511
agent_waits_total - Total agent waits monitor element 512
agents_created_empty_pool - Agents Created Due to Empty Agent Pool 513
agents_from_pool - Agents Assigned From Pool 513
agents_registered - Agents Registered 514
agents_registered_top - Maximum Number of Agents Registered 514
agents_stolen - Stolen Agents 514
agents_top - Number of Agents Created 515
agents_waiting_on_token - Agents Waiting for a Token 515
agents_waiting_top - Maximum Number of Agents Waiting monitor element 516

agg_temp_tablespace_top - Aggregate temporary table space top monitor element 516
aggsqltempspace_threshold_id - Aggregate SQL temporary space threshold ID monitor element . . 517
aggsqltempspace_threshold_value - AggSQL temporary space threshold value monitor element . 517
aggsqltempspace_threshold_violated - AggSQL temporary space threshold violated monitor element 517
app_rqsts_completed_total - Total application requests completed monitor element. 518
appl_con_time - Connection Request Start Timestamp 519
appl_id - Application ID monitor element 519
appl_id_holding_lk - Application ID Holding Lock 521
appl_id_oldest_xact - Application with Oldest Transaction 522
appl_idle_time - Application Idle Time 522
appl_name - Application name monitor element 523
appl_priority - Application Agent Priority 523
appl_priority_type - Application Priority Type . . 524
appl_section_inserts - Section Inserts monitor element 525
appl_section_lookups - Section Lookups 525
appl_status - Application Status 525
application_handle - Application handle monitor element 528
appls_cur_cons - Applications Connected Currently 529
appls_in_db2 - Applications Executing in the Database Currently 529
arm_correlator - Application response measurement correlator monitor element 530
associated_agents_top - Maximum Number of Associated Agents. 530
async_runstats – Total number of asynchronous RUNSTATS requests monitor element 530
audit_events_total - Total audit events monitor element 531
audit_file_write_wait_time - Audit file write wait time monitor element. 532
audit_file_writes_total - Total audit files written monitor element 533
audit_subsystem_wait_time - Audit subsystem wait time monitor element. 534
audit_subsystem_waits_total - Total audit subsystem waits monitor element 536
auth_id - Authorization ID 537
authority_bitmap - User authorization level monitor element 537
authority_lvl - User authorization level monitor element 538
auto_storage_hybrid - Hybrid automatic storage table space indicator monitor element 539
automatic - Buffer pool automatic monitor element 540
bin_id - Histogram bin identifier monitor element 540
binds_precompiles - Binds/Precompiles Attempted 540
block_ios - Number of block I/O requests monitor element 541
blocking_cursor - Blocking Cursor 542
blocks_pending_cleanup - Pending cleanup rolled-out blocks monitor element 542
bottom - Histogram bin bottom monitor element 542
boundary_leaf_node_splits - Boundary leaf node splits monitor element 543
bp_cur_buffsz - Current Size of Buffer Pool . . . 543
bp_id - Buffer pool identifier monitor element . . 543
bp_name - Buffer pool name monitor element . . 543
bp_new_buffsz - New Buffer Pool Size 544
bp_pages_left_to_remove - Number of Pages Left to Remove 544
bp_tbsp_use_count - Number of Table Spaces Mapped to Buffer Pool 544
buff_auto_tuning - FCM buffer auto-tuning indicator monitor element 544
buff_free - FCM buffers currently free monitor element 545
buff_free_bottom - Minimum FCM buffers free monitor element 545
buff_max - Maximum possible number of FCM buffers monitor element 546
buff_total - Number of currently allocated FCM buffers monitor element 546
byte_order - Byte Order of Event Data 547
cat_cache_inserts - Catalog cache inserts monitor element 547
cat_cache_lookups - Catalog cache lookups monitor element 548
cat_cache_overflows - Catalog Cache Overflows 550
cat_cache_size_top - Catalog cache high watermark monitor element 551
catalog_node - Catalog Node Number 551
catalog_node_name - Catalog Node Network Name 552
ch_auto_tuning - FCM channel auto-tuning indicator monitor element 552
ch_free - Channels currently free monitor element 552
ch_free_bottom - Minimum channels free monitor element 553
ch_max - Maximum possible number of FCM channels monitor element 553
ch_total - Number of currently allocated FCM channels monitor element 554
client_acctng - Client accounting string monitor element 554
client_applname - Client application name monitor element 555
client_db_alias - Database Alias Used by Application 556
client_hostname - Client hostname monitor element 557
client_idle_wait_time - Client idle wait time monitor element 557
client_pid - Client process ID monitor element . . 558
client_platform - Client operating platform monitor element 559
client_port_number - Client port number monitor element 560
client_prdid - Client product and version ID monitor element 560
client_protocol - Client communication protocol monitor element 561
client_userid - Client user ID monitor element . . 562

client_wrkstnname - Client workstation name monitor element 563
codepage_id - ID of Code Page Used by Application 564
comm_private_mem - Committed Private Memory 564
commit_sql_stmts - Commit Statements Attempted 565
comp_env_desc - Compilation environment monitor element 565
completion_status - Completion status monitor element 566
con_elapsed_time - Most Recent Connection Elapsed Time 567
con_local_dbases - Local Databases with Current Connects 567
con_response_time - Most Recent Response Time for Connect 567
concurrent_act_top - Concurrent activity top monitor element 568
concurrent_connection_top - Concurrent connection top monitor element 568
concurrent_wlo_act_top - Concurrent WLO activity top monitor element 569
concurrent_wlo_top - Concurrent workload occurrences top monitor element 569
concurrentdbcoordactivities_db_threshold_id - Concurrent database coordinator activities database threshold ID monitor element 569
concurrentdbcoordactivities_db_threshold _queued - Concurrent database coordinator activities database threshold queued monitor element . . . 570
concurrentdbcoordactivities_db_threshold _value - Concurrent database coordinator activities database threshold value monitor element 570
concurrentdbcoordactivities_db_threshold _violated - Concurrent database coordinator activities database threshold violated monitor element . . . 571
concurrentdbcoordactivities_subclass_threshold _id - Concurrent database coordinator activities service subclass threshold ID monitor element 571
concurrentdbcoordactivities_subclass_ threshold_queued - Concurrent database coordinator activities service subclass threshold queued monitor element 571
concurrentdbcoordactivities_subclass_ threshold_value - Concurrent database coordinator activities service subclass threshold value monitor element 572
concurrentdbcoordactivities_subclass_ threshold_violated - Concurrent database coordinator activities service subclass threshold violated monitor element 572
concurrentdbcoordactivities_superclass_ threshold_id - Concurrent database coordinator activities service superclass threshold ID monitor element 573
concurrentdbcoordactivities_superclass_ threshold_queued - Concurrent database coordinator activities service superclass threshold queued monitor element 573
concurrentdbcoordactivities_superclass_ threshold_value - Concurrent database coordinator activities service superclass threshold value monitor element 573
concurrentdbcoordactivities_superclass_ threshold_violated - Concurrent database coordinator activities service superclass threshold violated monitor element 574
concurrentdbcoordactivities_wl_was_threshold _id - Concurrent database coordinator activities workload work action set threshold ID monitor element 574
concurrentdbcoordactivities_wl_was_threshold _queued - Concurrent database coordinator activities workload work action set threshold queued monitor element 575
concurrentdbcoordactivities_wl_was_threshold _value - Concurrent database coordinator activities workload work action set threshold value monitor element 575
concurrentdbcoordactivities_wl_was_threshold _violated - Concurrent database coordinator activities workload work action set threshold violated monitor element 576
concurrentdbcoordactivities_work_action_set_ threshold_id - Concurrent database coordinator activities work action set threshold ID monitor element 576
concurrentdbcoordactivities_work_action_set_ threshold_queued - Concurrent database coordinator activities work action set threshold queued monitor element 577
concurrentdbcoordactivities_work_action_set_ threshold_value - Concurrent database coordinator activities work action set threshold value monitor element 577
concurrentdbcoordactivities_work_action_set_ threshold_violated - Concurrent database coordinator activities work action set threshold violated monitor element 578
conn_complete_time - Connection Request Completion Timestamp 578
conn_time - Time of database connection monitor element 578
connection_start_time - Connection start time monitor element 579
connection_status - Connection status monitor element 579
connections_top - Maximum Number of Concurrent Connections 580
consistency_token - Package consistency token monitor element 580
container_accessible - Accessibility of container monitor element 581
container_id - Container identification monitor element 581
container_name - Container name monitor element 581
container_stripe_set - Container stripe set monitor element 582
container_total_pages - Total pages in container monitor element 582

container_type - Container type monitor element 583
container_usable_pages - Usable pages in container monitor element 583
coord_act_aborted_total - Coordinator activities aborted total monitor element 583
coord_act_completed_total - Coordinator activities completed total monitor element 584
coord_act_est_cost_avg - Coordinator activity estimated cost average monitor element 584
coord_act_exec_time_avg - Coordinator activities execution time average monitor element 585
coord_act_interarrival_time_avg - Coordinator activity arrival time average monitor element . . 586
coord_act_lifetime_avg - Coordinator activity lifetime average monitor element. 587
coord_act_lifetime_top - Coordinator activity lifetime top monitor element 587
coord_act_queue_time_avg - Coordinator activity queue time average monitor element 588
coord_act_rejected_total - Coordinator activities rejected total monitor element 589
coord_agent_pid - Coordinator agent identifier monitor element 589
coord_agents_top - Maximum Number of Coordinating Agents 589
coord_member - Coordinator member monitor element 590
coord_node - Coordinating Node. 590
coord_partition_num - Coordinator partition number monitor element 591
coord_stmt_exec_time - Execution time for statement by coordinator agent monitor element . 591
corr_token - DRDA Correlation Token 591
cost_estimate_top - Cost estimate top monitor element 592
count - Number of Event Monitor Overflows. . . 592
cpu_configured - Number of configured CPUs monitor element 593
cpu_cores_per_socket - Number of CPU cores per socket monitor element 593
cpu_hmt_degree - Number of logical CPUs monitor element 593
cpu_idle - Processor idle time monitor element . . 593
cpu_iowait - IO Wait time monitor element . . . 594
cpu_load_long - Processor load (long timeframe) monitor element 595
cpu_load_medium - Processor load (medium timeframe) monitor element 595
cpu_load_short - Processor load (short timeframe) monitor element 595
cpu_online - Number of CPUs online monitor element 596
cpu_speed - CPU clock speed monitor element . . 596
cpu_system - Kernel time monitor element . . . 596
cpu_timebase - Frequency of timebase register increment monitor element. 597
cpu_total - Number of CPUs monitor element . . 597
cpu_usage_total - Processor usage monitor element 597
cpu_user - Non-kernel processing time monitor element 598
cputime_threshold_id - CPU time threshold ID monitor element 598
cputime_threshold_value - CPU time threshold value monitor element 599
cputime_threshold_violated - CPU time threshold violated monitor element 599
cputimeinsc_threshold_id - CPU time in service class threshold ID monitor element 599
cputimeinsc_threshold_value - CPU time in service class threshold value monitor element 600
cputimeinsc_threshold_violated - CPU time in service class threshold violated monitor element. . 600
create_nickname - Create Nicknames 600
create_nickname_time - Create Nickname Response Time 601
creator - Application Creator 601
current_active_log - Current Active Log File Number 602
current_archive_log - Current Archive Log File Number 602
current_extent - Extent currently being moved monitor element 603
cursor_name - Cursor Name 603
data_object_pages - Data Object Pages 603
data_object_l_pages - Table data logical pages monitor element 604
data_partition_id - Data partition identifier monitor element 604
datasource_name - Data Source Name 605
db2_process_id - DB2 process ID monitor element 605
db2_process_name - DB2 process name monitor element 606
db2_status - Status of DB2 Instance 606
db2start_time - Start Database Manager Timestamp 606
db_conn_time - Database activation timestamp monitor element 607
db_heap_top - Maximum Database Heap Allocated 607
db_location - Database Location 607
db_name - Database Name. 608
db_path - Database Path. 609
db_status - Status of Database. 609
db_storage_path - Automatic storage path monitor element 610
db_storage_path_state - Storage path state monitor element 610
db_storage_path_with_dpe - Storage path including database partition expression monitor element . . 611
db_work_action_set_id - Database work action set ID monitor element 611
db_work_class_id - Database work class ID monitor element 611
dcs_appl_status - DCS Application Status 612
dcs_db_name - DCS Database Name 612
ddl_sql_stmts - Data Definition Language (DDL) SQL Statements. 613
deadlock_id - Deadlock Event Identifier 613
deadlock_node - Partition Number Where Deadlock Occurred 614
deadlocks - Deadlocks detected monitor element 614
degree_parallelism - Degree of Parallelism. . . . 616

del_keys_cleaned - Pseudo deleted keys cleaned monitor element 617
delete_sql_stmts - Deletes 617
delete_time - Delete Response Time 617
destination_service_class_id – Destination service class ID monitor element 618
diaglog_write_wait_time - Diagnostic log file write wait time monitor element 618
diaglog_writes_total - Total diagnostic log file writes monitor element 619
direct_read_reqs - Direct read requests monitor element 620
direct_read_time - Direct read time monitor element 622
direct_reads - Direct reads from database monitor element 624
direct_write_reqs - Direct write requests monitor element 626
direct_write_time - Direct write time monitor element 627
direct_writes - Direct writes to database monitor element 629
disconn_time - Database Deactivation Timestamp 631
disconnects - Disconnects 631
dl_conns - Connections involved in deadlock monitor element 632
dynamic_sql_stmts - Dynamic SQL Statements Attempted 632
edu_ID - Engine dispatchable unit ID monitor element 633
eff_stmt_text - Effective statement text monitor element 633
effective_isolation - Effective isolation monitor element 633
effective_lock_timeout - Effective lock timeout monitor element 634
effective_query_degree - Effective query degree monitor element 634
elapsed_exec_time - Statement Execution Elapsed Time 634
empty_pages_deleted - Empty pages deleted monitor element 635
empty_pages_reused - Empty pages reused monitor element 635
entry_time - Entry time monitor element 636
estimatedsqlcost_threshold_id - Estimated SQL cost threshold ID monitor element 636
estimatedsqlcost_threshold_value - Estimated SQL cost threshold value monitor element 636
estimatedsqlcost_threshold_violated - Estimated SQL cost threshold violated monitor element . . . 637
event_monitor_name - Event Monitor Name . . . 637
event_time - Event Time. 637
evmon_activates - Number of Event Monitor Activations 637
evmon_flushes - Number of Event Monitor Flushes 638
executable_id - Executable ID monitor element . . 639
execution_id - User Login ID 639
failed_sql_stmts - Failed Statement Operations . . 640
fcm_message_recv_volume - FCM message received volume monitor element 640
fcm_message_recv_wait_time - FCM message received wait time monitor element 641
fcm_message_recvs_total - Total FCM message receives monitor element 643
fcm_message_send_volume - FCM message send volume monitor element 644
fcm_message_send_wait_time - FCM message send wait time monitor element 645
fcm_message_sends_total - Total FCM message sends monitor element 646
fcm_recv_volume - FCM received volume monitor element 647
fcm_recv_wait_time - FCM received wait time monitor element 648
fcm_recvs_total - FCM receives total monitor element 650
fcm_send_volume - FCM send volume monitor element 651
fcm_send_wait_time - FCM send wait time monitor element 652
fcm_sends_total - FCM sends total monitor element 654
fcm_tq_recv_volume - FCM table queue received volume monitor element 655
fcm_tq_recv_wait_time - FCM table queue received wait time monitor element 656
fcm_tq_recvs_total - FCM table queue receives total monitor element 657
fcm_tq_send_volume - FCM table queue send volume monitor element 658
fcm_tq_send_wait_time - FCM table queue send wait time monitor element 659
fcm_tq_sends_total - FCM table queue send total monitor element 660
fetch_count - Number of Successful Fetches . . . 661
files_closed - Database files closed monitor element 662
first_active_log - First Active Log File Number . . 663
first_overflow_time - Time of First Event Overflow 663
fs_caching - File system caching monitor element 664
fs_id - Unique file system identification number monitor element 664
fs_total_size - Total size of a file system monitor element 665
fs_used_size - Amount of space used on a file system monitor element 665
gw_comm_error_time - Communication Error Time 666
gw_comm_errors - Communication Errors. 666
gw_con_time - DB2 Connect Gateway First Connect Initiated 666
gw_connections_top - Maximum Number of Concurrent Connections to Host Database. . . . 666
gw_cons_wait_client - Number of Connections Waiting for the Client to Send Request 667
gw_cons_wait_host - Number of Connections Waiting for the Host to Reply 667
gw_cur_cons - Current Number of Connections for DB2 Connect 668
gw_db_alias - Database Alias at the Gateway. . . 668
gw_exec_time - Elapsed Time Spent on DB2 Connect Gateway Processing 668
gw_total_cons - Total Number of Attempted Connections for DB2 Connect 669

hadr_connect_status - HADR Connection Status monitor element 669
hadr_connect_time - HADR Connection Time monitor element 670
hadr_heartbeat - HADR Heartbeat monitor element 670
hadr_local_host - HADR Local Host monitor element 671
hadr_local_service - HADR Local Service monitor element 671
hadr_log_gap - HADR Log Gap 672
hadr_peer_window - HADR peer window monitor element 672
hadr_peer_window_end - HADR peer window end monitor element 672
hadr_primary_log_file - HADR Primary Log File monitor element 673
hadr_primary_log_lsn - HADR Primary Log LSN monitor element 673
hadr_primary_log_page - HADR Primary Log Page monitor element 674
hadr_remote_host - HADR Remote Host monitor element 674
hadr_remote_instance - HADR Remote Instance monitor element 674
hadr_remote_service - HADR Remote Service monitor element 675
hadr_role - HADR Role 675
hadr_standby_log_file - HADR Standby Log File monitor element 676
hadr_standby_log_lsn - HADR Standby Log LSN monitor element 676
hadr_standby_log_page - HADR Standby Log Page monitor element 676
hadr_state - HADR State monitor element. . . . 677
hadr_syncmode - HADR Synchronization Mode monitor element 677
hadr_timeout - HADR Timeout monitor element 678
hash_join_overflows - Hash Join Overflows . . . 678
hash_join_small_overflows - Hash Join Small Overflows 679
histogram_type - Histogram type monitor element 679
hld_application_handle - Identifier for the application holding the lock monitor element. . . 680
hld_member - Database member for application holding lock. 681
host_ccsid - Host Coded Character Set ID 681
host_db_name - Host Database Name 681
host_prdid - Host Product/Version ID 681
host_response_time - Host Response Time. . . . 682
hostname - Host name monitor element 683
host_name - Host name monitor element 683
ida_recv_volume - Total data volume received monitor element 683
ida_recv_wait_time - Time spent waiting to receive data monitor element. 685
ida_recvs_total - Number of times data received monitor element 686
ida_send_volume - Total data volume sent monitor element 687
ida_send_wait_time - Time spent waiting to send data monitor element. 689
ida_sends_total - Number of times data sent monitor element 690
idle_agents - Number of Idle Agents 692
iid - Index identifier monitor element 692
inbound_bytes_received - Inbound Number of Bytes Received 692
inbound_bytes_sent - Inbound Number of Bytes Sent 692
inbound_comm_address - Inbound Communication Address 693
include_col_updates - Include column updates monitor element 693
index_object_pages - Index Object Pages 693
index_object_l_pages - Index data logical pages monitor element 694
index_only_scans - Index-only scans monitor element 694
index_scans - Index scans monitor element . . . 694
index_tbsp_id - Index table space ID monitor element 694
input_db_alias - Input Database Alias 695
insert_sql_stmts - Inserts 695
insert_time - Insert Response Time 696
insert_timestamp - Insert timestamp monitor element 696
int_auto_rebinds - Internal Automatic Rebinds . . 697
int_commits - Internal commits monitor element 698
int_deadlock_rollbacks - Internal Rollbacks Due To Deadlock. 699
int_node_splits - Intermediate node splits monitor element 700
int_rollbacks - Internal rollbacks monitor element 700
int_rows_deleted - Internal Rows Deleted 702
int_rows_inserted - Internal Rows Inserted . . . 702
int_rows_updated - Internal Rows Updated . . . 703
invocation_id - Invocation ID monitor element . . 703
ipc_recv_volume - Interprocess communication received volume monitor element 704
ipc_recv_wait_time - Interprocess communication received wait time monitor element 705
ipc_recvs_total - Interprocess communication receives total monitor element. 706
ipc_send_volume - Interprocess communication send volume monitor element. 707
ipc_send_wait_time - Interprocess communication send wait time monitor element 708
ipc_sends_total - Interprocess communication send total monitor element. 709
is_system_appl - Is System Application monitor element 710
key_updates - Key updates monitor element . . . 710
last_active_log - Last Active Log File Number . . 711
last_backup - Last Backup Timestamp 711
last_executable_id - Last executable identifier monitor element 712
last_extent - Last extent moved monitor element 712
last_metrics_update - Metrics last update timestamp monitor element 712
last_overflow_time - Time of Last Event Overflow 712
last_reference_time - Last reference time monitor element 713

last_request_type - Last request type monitor element 713
last_reset - Last Reset Timestamp 714
last_wlm_reset - Time of last reset monitor element 714
lob_object_pages - LOB Object Pages 715
lob_object_l_pages - LOB data logical pages monitor element 715
local_cons - Local Connections 715
local_cons_in_exec - Local Connections Executing in the Database Manager 716
local_start_time - Local start time monitor element 717
lock_attributes - Lock attributes monitor element 717
lock_count - Lock count monitor element 718
lock_current_mode - Original lock mode before conversion monitor element 719
lock_escalation - Lock escalation monitor element 720
lock_escals - Number of lock escalations monitor element 721
lock_hold_count - Lock hold count monitor element 723
lock_list_in_use - Total lock list memory in use monitor element 723
lock_mode - Lock mode monitor element 724
lock_mode_requested - Lock mode requested monitor element 725
lock_name - Lock name monitor element 726
lock_node - Lock Node 727
lock_object_name - Lock Object Name 727
lock_object_type - Lock object type waited on monitor element 728
lock_release_flags - Lock release flags monitor element 730
lock_status - Lock status monitor element 730
lock_timeout_val - Lock timeout value monitor element 731
lock_timeouts - Number of lock timeouts monitor element 732
lock_wait_end_time - Lock wait end timestamp monitor element 733
lock_wait_start_time - Lock wait start timestamp monitor element 734
lock_wait_time - Time waited on locks monitor element 734
lock_wait_time_top – Lock wait time top monitor element 736
lock_waits - Lock waits monitor element 737
locks_held - Locks held monitor element 738
locks_held_top - Maximum number of locks held monitor element 739
locks_in_list - Number of Locks Reported 740
locks_waiting - Current agents waiting on locks monitor element 740
log_buffer_wait_time - Log buffer wait time monitor element 740
log_disk_wait_time - Log disk wait time monitor element 741
log_disk_waits_total - Total log disk waits monitor element 743
log_held_by_dirty_pages - Amount of Log Space Accounted for by Dirty Pages 744
log_read_time - Log Read Time 744
log_reads - Number of Log Pages Read 745
log_to_redo_for_recovery - Amount of Log to be Redone for Recovery 745
log_write_time - Log Write Time 746
log_writes - Number of Log Pages Written . . . 746
long_object_pages - Long Object Pages 747
long_object_l_pages - Long object data logical pages monitor element 747
long_tbsp_id - Long table space ID monitor element 747
machine_identification - Host hardware identification monitor element. 748
max_agent_overflows - Maximum Agent Overflows 748
max_data_received_1024 - Number of Statements with Outbound Bytes Received Between 513 and 1024 Bytes 748
max_data_received_128 - Number of Statements with Outbound Bytes Received Between 1 and 128 Bytes 749
max_data_received_16384 - Number of Statements with Outbound Bytes Received Between 8193 and 16384 Bytes 749
max_data_received_2048 - Number of Statements with Outbound Bytes Received Between 1025 and 2048 Bytes 750
max_data_received_256 - Number of Statements with Outbound Bytes Received Between 129 and 256 Bytes. 750
max_data_received_31999 - Number of Statements with Outbound Bytes Received Between 16385 and 31999 Bytes monitor element 751
max_data_received_4096 - Number of Statements with Outbound Bytes Received Between 2049 and 4096 Bytes 751
max_data_received_512 - Number of Statements with Outbound Bytes Received Between 257 and 512 Bytes. 751
max_data_received_64000 - Number of Statements with Outbound Bytes Received Between 32000 and 64000 Bytes monitor element 752
max_data_received_8192 - Number of Statements with Outbound Bytes Received Between 4097 and 8192 Bytes 752
max_data_received_gt64000 - Number of Statements with Outbound Bytes Received Greater than 64000 Bytes 753
max_data_sent_1024 - Number of Statements with Outbound Bytes Sent Between 513 and 1024 Bytes . 753
max_data_sent_128 - Number of Statements with Outbound Bytes Sent Between 1 and 128 Bytes . . 753
max_data_sent_16384 - Number of Statements with Outbound Bytes Sent Between 8193 and 16384 Bytes 754
max_data_sent_2048 - Number of Statements with Outbound Bytes Sent Between 1025 and 2048 Bytes 754
max_data_sent_256 - Number of Statements with Outbound Bytes Sent Between 129 and 256 Bytes . 755
max_data_sent_31999 - Number of Statements with Outbound Bytes Sent Between 16385 and 31999 Bytes 755

max_data_sent_4096 - Number of Statements with Outbound Bytes Sent Between 2049 and 4096 Bytes 756
max_data_sent_512 - Number of Statements with Outbound Bytes Sent Between 257 and 512 Bytes . 756
max_data_sent_64000 - Number of Statements with Outbound Bytes Sent Between 32000 and 64000 Bytes 756
max_data_sent_8192 - Number of Statements with Outbound Bytes Sent Between 4097 and 8192 Bytes 757
max_data_sent_gt64000 - Number of Statements with Outbound Bytes Sent Greater than 64000 Bytes 757
max_network_time_100_ms - Number of Statements with Network Time between 16 and 100 ms 758
max_network_time_16_ms - Number of Statements with Network Time between 4 and 16 ms 758
max_network_time_1_ms - Number of Statements with Network Time of up to 1 ms 759
max_network_time_4_ms - Number of Statements with Network Time between 1 and 4 ms 759
max_network_time_500_ms - Number of Statements with Network Time between 100 and 500 ms 759
max_network_time_gt500_ms - Number of Statements with Network Time greater than 500 ms 760
member - Database member monitor element . . 760
memory_free - Amount of free physical memory monitor element 762
memory_pool_id - Memory pool identifier monitor element 762
memory_pool_type - Memory pool name monitor element 762
memory_pool_used - Amount of memory pool in use monitor element 764
memory_pool_used_hwm - Memory pool high water mark monitor element 764
memory_set_committed - Memory currently committed monitor element 765
memory_set_id - Memory set identifier monitor element 765
memory_set_size - Memory set size monitor element 765
memory_set_type - Memory set type monitor element 765
memory_set_used - Memory in use by this set monitor element 766
memory_set_used_hwm - Memory set high water mark monitor element 766
memory_swap_free - Total free swap space monitor element 766
memory_swap_total - Total swap space monitor element 767
memory_total - Total physical memory monitor element 767
message - Control Table Message 767
message_time - Timestamp Control Table Message 768
nesting_level - Nesting level monitor element . . 768
network_time_bottom - Minimum Network Time for Statement 768
network_time_top - Maximum Network Time for Statement 769
nleaf - Number of leaf pages monitor element . . 770
nlevels - Number of index levels monitor element 770
node_number - Node Number 770
nonboundary_leaf_node_splits - Non-boundary leaf node splits monitor element 771
num_agents - Number of Agents Working on a Statement 771
num_assoc_agents - Number of Associated Agents 771
num_compilations - Statement Compilations . . . 772
num_coord_exec - Number of executions by coordinator agent monitor element 772
num_coord_exec_with_metrics - Number of executions by coordinator agent with metrics monitor element 772
num_db_storage_paths - Number of automatic storage paths 773
num_exec_with_metrics - Number of executions with metrics collected monitor element 773
num_executions - Statement executions monitor element 773
num_extents_left - Number of extents left to process monitor element. 774
num_extents_moved - Number of extents moved monitor element 774
num_gw_conn_switches - Connection Switches . . 775
num_indoubt_trans - Number of Indoubt Transactions 775
num_log_buffer_full - Number of times full log buffer caused agents to wait monitor element . . 775
num_log_data_found_in_buffer - Number of Log Data Found In Buffer. 777
num_log_part_page_io - Number of Partial Log Page Writes 777
num_log_read_io - Number of Log Reads 778
num_log_write_io - Number of Log Writes . . . 778
num_lw_thresh_exceeded - Number of lock wait thresholds exceeded monitor element 778
num_nodes_in_db2_instance - Number of Nodes in Partition 780
num_remaps - Number of remaps monitor element 780
num_threshold_violations - Number of threshold violations monitor element 780
num_transmissions - Number of Transmissions . . 781
num_transmissions_group - Number of Transmissions Group 781
number_in_bin - Number in bin monitor element 782
olap_func_overflows - OLAP Function Overflows monitor element 782
open_cursors - Number of Open Cursors 782
open_loc_curs - Open Local Cursors. 783
open_loc_curs_blk - Open Local Cursors with Blocking 783
open_rem_curs - Open Remote Cursors 784
open_rem_curs_blk - Open Remote Cursors with Blocking 784
os_level - Operating system level monitor element 785
os_name - Operating system name monitor element 785

os_release - Operating system release monitor element 785
os_version - Operating system version monitor element 785
outbound_appl_id - Outbound Application ID . . 785
outbound_bytes_received - Outbound Number of Bytes Received 786
outbound_bytes_received_bottom - Minimum Outbound Number of Bytes Received 787
outbound_bytes_received_top - Maximum Outbound Number of Bytes Received 787
outbound_bytes_sent - Outbound Number of Bytes Sent 787
outbound_bytes_sent_bottom - Minimum Outbound Number of Bytes Sent 788
outbound_bytes_sent_top - Maximum Outbound Number of Bytes Sent 788
outbound_comm_address - Outbound Communication Address 788
outbound_comm_protocol - Outbound Communication Protocol 789
outbound_sequence_no - Outbound Sequence Number 789
overflow_accesses - Accesses to overflowed records monitor element 789
overflow_creates - Overflow creates monitor element 790
package_id - Package identifier monitor element 790
package_elapsed_time - Package elapsed time monitor element 790
package_list_count - Package list count monitor element 791
package_list_exceeded - Package list exceeded monitor element 791
package_name - Package name monitor element 791
package_schema - Package schema monitor element 792
package_version_id - Package version monitor element 792
packet_receive_errors - Packet receive errors monitor element 793
packets_received - Packets received monitor element 793
packet_send_errors - Packet send errors monitor element 793
packets_sent - Packets sent monitor element . . . 794
page_allocations - Page allocations monitor element 794
page_reorgs - Page reorganizations monitor element 794
pages_from_block_ios - Total number of pages read by block I/O monitor element. 795
pages_from_vectored_ios - Total number of pages read by vectored I/O monitor element 796
pages_merged - Pages merged monitor element 796
pages_read - Number of pages read monitor element 796
pages_written - Number of pages written monitor element 796
parent_activity_id - Parent activity ID monitor element 797
parent_uow_id - Parent unit of work ID monitor element 797
partial_record - Partial Record monitor element 798
participant_no - Participant within Deadlock . . . 798
participant_no_holding_lk - Participant Holding a Lock on the Object Required by Application . . . 799
partition_number - Partition Number 799
passthru_time - Pass-Through Time 799
passthrus - Pass-Through 800
piped_sorts_accepted - Piped Sorts Accepted . . . 800
piped_sorts_requested - Piped Sorts Requested . . 801
pkg_cache_inserts - Package cache inserts monitor element 801
pkg_cache_lookups - Package cache lookups monitor element 802
pkg_cache_num_overflows - Package Cache Overflows 804
pkg_cache_size_top - Package cache high watermark 805
pool_async_data_read_reqs - Buffer pool asynchronous read requests monitor element. . . . 806
pool_async_data_reads - Buffer pool asynchronous data reads monitor element. 807
pool_async_data_writes - Buffer pool asynchronous data writes monitor element 808
pool_async_index_read_reqs - Buffer pool asynchronous index read requests monitor element. 808
pool_async_index_reads - Buffer pool asynchronous index reads monitor element . . . 809
pool_async_index_writes - Buffer pool asynchronous index writes monitor element . . . 810
pool_async_read_time - Buffer Pool Asynchronous Read Time 811
pool_async_write_time - Buffer pool asynchronous write time monitor element. 812
pool_async_xda_read_reqs - Buffer pool asynchronous XDA read requests monitor element . 812
pool_async_xda_reads - Buffer pool asynchronous XDA data reads monitor element. 813
pool_async_xda_writes - Buffer pool asynchronous XDA data writes monitor element 814
pool_config_size - Configured Size of Memory Pool 815
pool_cur_size - Current Size of Memory Pool. . . 815
pool_data_l_reads - Buffer pool data logical reads monitor element 816
pool_data_p_reads - Buffer pool data physical reads monitor element 818
pool_data_writes - Buffer pool data writes monitor element 820
pool_drty_pg_steal_clns - Buffer pool victim page cleaners triggered monitor element 822
pool_drty_pg_thrsh_clns - Buffer pool threshold cleaners triggered monitor element 823
pool_id - Memory Pool Identifier. 824
pool_index_l_reads - Buffer pool index logical reads monitor element 825
pool_index_p_reads - Buffer pool index physical reads monitor element 827
pool_index_writes - Buffer pool index writes monitor element 829

pool_lsn_gap_clns - Buffer pool log space cleaners triggered monitor element 831
pool_no_victim_buffer - Buffer pool no victim buffers monitor element 832
pool_read_time - Total buffer pool physical read time monitor element. 833
pool_secondary_id - Memory Pool Secondary Identifier 834
pool_temp_data_l_reads - Buffer pool temporary data logical reads monitor element 835
pool_temp_data_p_reads - Buffer pool temporary data physical reads monitor element 837
pool_temp_index_l_reads - Buffer pool temporary index logical reads monitor element 839
pool_temp_index_p_reads - Buffer pool temporary index physical reads monitor element 841
pool_temp_xda_l_reads - Buffer pool temporary XDA data logical reads monitor element 843
pool_temp_xda_p_reads - Buffer pool temporary XDA data physical reads monitor element. . . . 845
pool_watermark - Memory Pool Watermark . . . 847
pool_write_time - Total buffer pool physical write time monitor element. 847
pool_xda_l_reads - Buffer pool XDA data logical reads monitor element 849
pool_xda_p_reads - Buffer pool XDA data physical reads monitor element 851
pool_xda_writes - Buffer pool XDA data writes monitor element 853
post_shrthreshold_hash_joins - Post threshold hash joins 855
post_shrthreshold_sorts - Post shared threshold sorts monitor element 856
post_threshold_hash_joins - Hash Join Threshold 857
post_threshold_olap_funcs - OLAP Function Threshold monitor element. 858
post_threshold_sorts - Post threshold sorts monitor element 858
prefetch_wait_time - Time waited for prefetch monitor element 860
prep_time - Preparation time monitor element . . 860
prep_time_best - Statement best preparation time monitor element 861
prep_time_worst - Statement worst preparation time monitor element. 861
prev_uow_stop_time - Previous Unit of Work Completion Timestamp 861
priv_workspace_num_overflows - Private Workspace Overflows 862
priv_workspace_section_inserts - Private Workspace Section Inserts 863
priv_workspace_section_lookups - Private Workspace Section Lookups 863
priv_workspace_size_top - Maximum Private Workspace Size. 864
product_name - Product Name 865
progress_completed_units - Completed Progress Work Units 865
progress_description - Progress Description . . . 865
progress_list_attr - Current Progress List Attributes 866
progress_list_cur_seq_num - Current Progress List Sequence Number. 866
progress_seq_num - Progress Sequence Number 866
progress_start_time - Progress Start Time 867
progress_total_units - Total Progress Work Units 867
progress_work_metric - Progress Work Metric . . 867
pseudo_deletes - Pseudo deletes monitor element 868
pseudo_empty_pages - Pseudo empty pages monitor element 868
qp_query_id - Query patroller query ID monitor element 868
query_card_estimate - Query Number of Rows Estimate 868
query_cost_estimate - Query cost estimate monitor element 869
queue_assignments_total - Queue assignments total monitor element 870
queue_size_top - Queue size top monitor element 870
queue_time_total - Queue time total monitor element 870
quiescer_agent_id - Quiescer Agent Identification 871
quiescer_auth_id - Quiescer User Authorization Identification 871
quiescer_obj_id - Quiescer Object Identification . . 871
quiescer_state - Quiescer State. 872
quiescer_ts_id - Quiescer Table Space Identification 872
range_adjustment - Range Adjustment 872
range_container_id - Range Container 873
range_end_stripe - End Stripe 873
range_max_extent - Maximum Extent in Range . . 873
range_max_page_number - Maximum Page in Range 873
range_num_containers - Number of Containers in Range 873
range_number - Range Number 874
range_offset - Range Offset. 874
range_start_stripe - Start Stripe 874
range_stripe_set_number - Stripe Set Number . . 874
reclaimable_space_enabled - Reclaimable space enabled indicator monitor element 874
rej_curs_blk - Rejected Block Cursor Requests . . 875
rem_cons_in - Remote Connections To Database Manager 875
rem_cons_in_exec - Remote Connections Executing in the Database Manager 876
remote_lock_time - Remote Lock Time 876
remote_locks - Remote Locks 877
remote_member - Remote member monitor element 877
reorg_completion - Reorganization Completion Flag 877
reorg_current_counter - Reorganize Progress . . . 878
reorg_end - Table Reorganize End Time 878
reorg_index_id - Index Used to Reorganize the Table 878
reorg_long_tbspc_id - Table Space Where Long Objects are Reorganized monitor element 878
reorg_max_counter - Total Amount of Reorganization 879
reorg_max_phase - Maximum Reorganize Phase 879

reorg_phase - Table reorganization phase monitor element 879
reorg_phase_start - Reorganize Phase Start Time 880
reorg_rows_compressed - Rows Compressed . . . 880
reorg_rows_rejected_for_compression - Rows Rejected for Compression 880
reorg_start - Table Reorganize Start Time 881
reorg_status - Table Reorganize Status 881
reorg_tbspc_id - Table Space Where Table or Data partition is Reorganized 881
reorg_type - Table Reorganize Attributes 882
reorg_xml_regions_compressed – XML regions compressed monitor element 882
reorg_xml_regions_rejected_for_compression – XML regions rejected for compression monitor element 883
req_agent_tid - Thread identifier for agent waiting to acquire lock monitor element 883
req_application_handle - Identifier for application waiting to acquire lock monitor element 883
req_executable_id - Identifier for statement section waiting to acquire lock monitor element 883
req_member - Member of application waiting to acquire lock monitor element 884
request_exec_time_avg - Request execution time average monitor element 884
rf_log_num - Log Being Rolled Forward 884
rf_status - Log Phase 885
rf_timestamp - Rollforward Timestamp 885
rf_type - Rollforward Type 885
rollback_sql_stmts - Rollback Statements Attempted 886
rolled_back_agent_id - Rolled Back Agent 886
rolled_back_appl_id - Rolled Back Application . . 887
rolled_back_participant_no - Rolled back application participant monitor element 887
rolled_back_sequence_no - Rolled Back Sequence Number 888
root_node_splits - Root node splits monitor element 888
routine_id - Routine ID monitor element 888
rows_deleted - Rows deleted monitor element . . 889
rows_fetched - Rows fetched monitor element . . 889
rows_inserted - Rows inserted monitor element 890
rows_modified - Rows modified monitor element 890
rows_read - Rows read monitor element 892
rows_returned - Rows returned monitor element 893
rows_returned_top - Actual rows returned top monitor element 895
rows_selected - Rows Selected 895
rows_updated - Rows updated monitor element 896
rows_written - Rows Written 897
rqsts_completed_total - Total requests completed monitor element 897
sc_work_action_set_id - Service class work action set ID monitor element 898
sc_work_class_id - Service class work class ID monitor element 899
sec_log_used_top - Maximum Secondary Log Space Used 899
sec_logs_allocated - Secondary Logs Allocated Currently 900
section_actuals - Section actuals monitor element 900
section_env - Section environment monitor element 901
section_number - Section number monitor element 901
section_type - Section type indicator monitor element 902
select_sql_stmts - Select SQL Statements Executed 902
select_time - Query Response Time 903
sequence_no - Sequence number monitor element 904
sequence_no_holding_lk - Sequence Number Holding Lock 904
server_db2_type - Database Manager Type at Monitored (Server) Node 905
server_instance_name - Server Instance Name . . 905
server_platform - Server Operating System . . . 906
server_prdid - Server Product/Version ID 906
server_version - Server Version 907
service_class_id - Service class ID monitor element 907
service_level - Service Level 908
service_subclass_name - Service subclass name monitor element 908
service_superclass_name - Service superclass name monitor element 909
session_auth_id - Session authorization ID monitor element 910
shr_workspace_num_overflows - Shared Workspace Overflows 911
shr_workspace_section_inserts - Shared Workspace Section Inserts 911
shr_workspace_section_lookups - Shared Workspace Section Lookups 912
shr_workspace_size_top - Maximum Shared Workspace Size 912
smallest_log_avail_node - Node with Least Available Log Space 913
sort_heap_allocated - Total Sort Heap Allocated 914
sort_heap_top - Sort private heap high watermark 914
sort_overflows - Sort overflows monitor element 915
sort_shrheap_allocated - Sort Share Heap Currently Allocated 916
sort_shrheap_top - Sort share heap high watermark 917
source_service_class_id - Source service class ID monitor element 917
sp_rows_selected - Rows Returned by Stored Procedures 918
sql_chains - Number of SQL Chains Attempted . . 918
sql_req_id - Request Identifier for SQL Statement 919
sql_reqs_since_commit - SQL Requests Since Last Commit 919
sql_stmts - Number of SQL Statements Attempted 919
sqlca - SQL Communications Area (SQLCA) . . . 920
sqlrowsread_threshold_id - SQL rows read threshold ID monitor element 920
sqlrowsread_threshold_value - SQL rows read threshold value monitor element 921
sqlrowsread_threshold_violated - SQL rows read threshold violated monitor element 921
sqlrowsreadinsc_threshold_id - SQL rows read in service class threshold ID monitor element . . . 921

sqlrowsreadinsc_threshold_value - SQL rows read in service class threshold value monitor element . 922
sqlrowsreadinsc_threshold_violated - SQL rows read in service class threshold violated monitor element 922
sqlrowsreturned_threshold_id - SQL rows read returned threshold ID monitor element 922
sqlrowsreturned_threshold_value - SQL rows read returned threshold value monitor element 923
sqlrowsreturned_threshold_violated - SQL rows read returned threshold violated monitor element . 923
sqltempspace_threshold_id - SQL temporary space threshold ID monitor element 923
sqltempspace_threshold_value - SQL temporary space threshold value monitor element 924
sqltempspace_threshold_violated - SQL temporary space threshold violated monitor element 924
ss_exec_time - Subsection Execution Elapsed Time 924
ss_node_number - Subsection Node Number . . . 925
ss_number - Subsection number monitor element 925
ss_status - Subsection Status 925
ss_sys_cpu_time - System CPU Time used by Subsection 926
ss_usr_cpu_time - User CPU Time used by Subsection 926
start_time - Event Start Time 927
static_sql_stmts - Static SQL Statements Attempted 927
statistics_timestamp - Statistics timestamp monitor element 928
stats_cache_size – Size of statistics cache monitor element 928
stats_fabricate_time – Total time spent on statistics fabrication activities monitor element 929
stats_fabrications – Total number of statistics fabrications monitor elements 929
status_change_time - Application Status Change Time 930
stmt_elapsed_time - Most Recent Statement Elapsed Time 930
stmt_exec_time - Statement execution time monitor element 931
stmt_first_use_time - Statement first use timestamp monitor element 931
stmt_history_id - Statement history identifier . . . 932
inact_stmthist_sz - Statement history list size . . . 932
stmt_invocation_id - Statement invocation identifier monitor element 932
stmt_isolation - Statement isolation 933
stmt_last_use_time - Statement last use timestamp monitor element 934
stmt_lock_timeout - Statement lock timeout monitor element 934
stmt_nest_level - Statement nesting level monitor element 935
stmt_node_number - Statement Node 935
stmt_operation/operation - Statement operation monitor element 935
stmt_pkgcache_id - Statement package cache identifier monitor element 937
stmt_query_id - Statement query identifier monitor element 938
stmt_sorts - Statement Sorts 938
stmt_source_id - Statement source identifier . . . 939
stmt_start - Statement Operation Start Timestamp 939
stmt_stop - Statement Operation Stop Timestamp 939
stmt_sys_cpu_time - System CPU Time used by Statement 940
stmt_text - SQL statement text monitor element 940
stmt_type - Statement type monitor element . . . 942
stmt_type_id - Statement type identifier monitor element 942
stmt_usr_cpu_time - User CPU Time used by Statement 943
stmt_value_data - Value data 944
stmt_value_index - Value index 944
stmt_value_isnull - Value has null value monitor element 945
stmt_value_isreopt - Variable used for statement reoptimization monitor element 945
stmt_value_type - Value type monitor element . . 946
sto_path_free_sz - Automatic storage path free space monitor element 946
stop_time - Event Stop Time 947
stored_proc_time - Stored Procedure Time 947
stored_procs - Stored Procedures 947
swap_pages_in - Pages swapped in from disk monitor element 948
swap_pages_out - Pages swapped out to disk monitor element 948
swap_page_size - Swap page size monitor element 948
sync_runstats – Total number of synchronous RUNSTATS activities monitor element 948
sync_runstats_time – Total time spent on synchronous RUNSTATS activities monitor element 949
system_auth_id - System authorization identifier monitor element 950
system_cpu_time - System CPU time monitor element 950
tab_file_id - Table file ID monitor element 951
tab_type - Table type monitor element 951
table_file_id - Table file ID monitor element . . . 951
table_name - Table name monitor element 952
table_scans - Table scans monitor element 953
table_schema - Table schema name monitor element 953
table_type - Table type monitor element 954
tablespace_auto_resize_enabled - Table space automatic resizing enabled monitor element . . . 955
tablespace_content_type - Table space content type monitor element 956
tablespace_cur_pool_id - Buffer pool currently being used monitor element 956
tablespace_current_size - Current table space size 956
tablespace_extent_size - Table space extent size monitor element 957
tablespace_free_pages - Free pages in table space monitor element 957
tablespace_id - Table space identification monitor element 958
tablespace_increase_size - Increase size in bytes 958
tablespace_increase_size_percent - Increase size by percent monitor element. 959

tablespace_initial_size - Initial table space size . . 959
tablespace_last_resize_failed - Last resize attempt failed 959
tablespace_last_resize_time - Time of last successful resize 960
tablespace_max_size - Maximum table space size 960
tablespace_min_recovery_time - Minimum recovery time for rollforward monitor element 960
tablespace_name - Table space name monitor element 961
tablespace_next_pool_id - Buffer pool that will be used at next startup monitor element 962
tablespace_num_containers - Number of Containers in Table Space 962
tablespace_num_quiescers - Number of Quiescers 962
tablespace_num_ranges - Number of Ranges in the Table Space Map 963
tablespace_page_size - Table space page size monitor element 963
tablespace_page_top - Table space high watermark monitor element 963
tablespace_paths_dropped - Table space using dropped path monitor element 964
tablespace_pending_free_pages - Pending free pages in table space monitor element 964
tablespace_prefetch_size - Table space prefetch size monitor element 965
tablespace_rebalancer_extents_processed - Number of Extents the Rebalancer has Processed 965
tablespace_rebalancer_extents_remaining - Total Number of Extents to be Processed by the Rebalancer 965
tablespace_rebalancer_last_extent_moved - Last Extent Moved by the Rebalancer 966
tablespace_rebalancer_mode - Rebalancer mode monitor element 966
tablespace_rebalancer_priority - Current Rebalancer Priority 968
tablespace_rebalancer_restart_time - Rebalancer Restart Time. 968
tablespace_rebalancer_start_time - Rebalancer Start Time 968
tablespace_state - Table space state monitor element 969
tablespace_state_change_object_id - State Change Object Identification 971
tablespace_state_change_ts_id - State Change Table Space Identification 971
tablespace_total_pages - Total pages in table space monitor element 971
tbsp_trackmod_state - Table space trackmod state monitor element 972
tablespace_type - Table space type monitor element 973
tablespace_usable_pages - Usable pages in table space monitor element 973
tablespace_used_pages - Used pages in table space monitor element 974
tablespace_using_auto_storage - Table space enabled for automatic storage monitor element . . 974
tbsp_max_page_top - Maximum table space page high watermark monitor element. 975
tcpip_recv_volume - TCP/IP received volume monitor element 975
tcpip_recv_wait_time - TCP/IP received wait time monitor element 976
tcpip_recvs_total - TCP/IP receives total monitor element 977
tcpip_send_volume - TCP/IP send volume monitor element 978
tcpip_send_wait_time - TCP/IP send wait time monitor element 979
tcpip_sends_total - TCP/IP sends total monitor element 980
temp_tablespace_top - Temporary table space top monitor element 980
territory_code - Database Territory Code 981
thresh_violations - Number of threshold violations monitor element 981
threshold_action - Threshold action monitor element 983
threshold_domain - Threshold domain monitor element 983
threshold_maxvalue - Threshold maximum value monitor element 984
threshold_name - Threshold name monitor element 984
threshold_predicate - Threshold predicate monitor element 984
threshold_queuesize - Threshold queue size monitor element 985
thresholdid - Threshold ID monitor element . . . 985
time_completed - Time completed monitor element 985
time_created - Time created monitor element . . . 986
time_of_violation - Time of violation monitor element 986
time_stamp - Snapshot Time 986
time_started - Time started monitor element . . . 986
time_zone_disp - Time Zone Displacement . . . 987
top - Histogram bin top monitor element 987
tot_log_used_top - Maximum Total Log Space Used 987
total_act_time - Total activity time monitor element 988
total_act_wait_time - Total activity wait time monitor element 989
total_app_commits - Total application commits monitor elements 990
total_app_rollbacks - Total application rollbacks monitor element 991
total_app_rqst_time - Total application request time monitor element 992
total_app_section_executions - Total application section executions monitor element 993
total_buffers_rcvd - Total FCM buffers received monitor element 994
total_buffers_sent - Total FCM buffers sent monitor element 994
total_bytes_received - Bytes received monitor element 995
total_bytes_sent - Bytes sent monitor element . . 995
total_commit_proc_time - Total commits processing time monitor element. 995
total_commit_time - Total commit time monitor element 996

total_compilations - Total compilations monitor element 997
total_compile_proc_time - Total compile processing time monitor element. 998
total_compile_time - Total compile time monitor element 999
total_cons - Connects Since Database Activation 1000
total_cpu_time - Total CPU time monitor element 1001
total_exec_time - Elapsed statement execution time monitor element 1002
total_hash_joins - Total Hash Joins 1002
total_hash_loops - Total Hash Loops 1003
total_implicit_compilations - Total implicit complications monitor element 1003
total_implicit_compile_proc_time - Total implicit compile processing time monitor element . . . 1004
total_implicit_compile_time - Total implicit compile time monitor element 1005
total_load_proc_time - Total load processing time monitor element 1006
total_load_time - Total load time monitor element 1007
total_loads - Total loads monitor element. 1008
total_log_available - Total Log Available 1009
total_log_used - Total Log Space Used. 1010
total_move_time - Total extent move time monitor element 1010
total_olap_funcs - Total OLAP Functions monitor element 1010
total_reorg_proc_time - Total reorganization processing time monitor element 1011
total_reorg_time - Total reorganization time monitor element 1012
total_reorgs - Total reorganizations monitor element 1013
total_rollback_proc_time - Total rollback processing time monitor element 1014
total_rollback_time - Total rollback time monitor element 1015
total_routine_invocations - Total routine invocations monitor elements 1016
total_routine_non_sect_proc_time - Non-section processing time monitor element 1017
total_routine_non_sect_time - Non-section routine execution time monitor elements 1017
total_routine_time - Total routine time monitor element 1018
total_routine_user_code_proc_time - Total routine user code processing time monitor element . . . 1020
total_routine_user_code_time - Total routine user code time monitor element 1021
total_rqst_mapped_in - Total request mapped-in monitor element 1022
total_rqst_mapped_out - Total request mapped-out monitor element. 1022
total_rqst_time - Total request time monitor element 1023
total_runstats - Total runtime statistics monitor element 1024
total_runstats_proc_time - Total runtime statistics processing time monitor element 1025
total_runstats_time - Total runtime statistics time monitor element 1026
total_sec_cons - Secondary Connections 1027
total_section_proc_time - Total section processing time monitor element 1027
total_section_sort_proc_time - Total section sort processing time monitor element 1028
total_section_sort_time - Total section sort time monitor element 1030
total_section_sorts - Total section sorts monitor element 1032
total_section_time - Total section time monitor element 1033
total_sort_time - Total sort time monitor element 1035
total_sorts - Total sorts monitor element 1036
total_sys_cpu_time - Total system CPU time for a statement monitor element 1037
total_usr_cpu_time - Total user CPU time for a statement monitor element 1038
total_wait_time - Total wait time monitor element 1038
tpmon_acc_str - TP monitor client accounting string monitor element 1039
tpmon_client_app - TP monitor client application name monitor element 1040
tpmon_client_userid - TP monitor client user ID monitor element 1040
tpmon_client_wkstn - TP monitor client workstation name monitor element. 1041
tq_cur_send_spills - Current number of table queue buffers overflowed monitor element . . . 1042
tq_id_waiting_on - Waited on node on a table queue monitor element. 1042
tq_max_send_spills - Maximum number of table queue buffers overflows 1042
tq_node_waited_for - Waited for node on a table queue 1043
tq_rows_read - Number of Rows Read from table queues 1043
tq_rows_written - Number of rows written to table queues 1044
tq_tot_send_spills - Total number of table queue buffers overflowed monitor element 1044
tq_wait_for_any - Waiting for any node to send on a table queue 1045
ts_name - Table space being rolled forward monitor element 1046
uid_sql_stmts - Update/Insert/Delete SQL Statements Executed. 1046
unread_prefetch_pages - Unread prefetch pages monitor element 1047
uow_comp_status - Unit of Work Completion Status 1047
uow_elapsed_time - Most Recent Unit of Work Elapsed Time 1048
uow_id - Unit of work ID monitor element . . . 1048
uow_lock_wait_time - Total time unit of work waited on locks monitor element 1049
uow_log_space_used - Unit of work log space used monitor element 1049
uow_start_time - Unit of work start timestamp monitor element 1050

uow_status - Unit of Work Status 1051
uow_stop_time - Unit of work stop timestamp monitor element 1051
uow_total_time_top - UOW total time top monitor element 1052
update_sql_stmts - Updates 1052
update_time - Update Response Time 1053
user_cpu_time - User CPU time monitor element 1053
utility_dbname - Database Operated on by Utility 1054
utility_description - Utility Description 1054
utility_id - Utility ID 1054
utility_invoker_type - Utility Invoker Type . . . 1055
utility_priority - Utility Priority 1055
utility_start_time - Utility Start Time 1055
utility_state - Utility State 1055
utility_type - Utility Type 1056
valid - Section validity indicator monitor element 1056
vectored_ios - Number of vectored I/O requests monitor element 1057
version - Version of Monitor Data 1057
virtual_mem_free - Free virtual memory monitor element 1057
virtual_mem_reserved - Reserved virtual memory monitor element 1058
virtual_mem_total - Total virtual memory monitor element 1058
wl_work_action_set_id - Workload work action set identifier monitor element 1058
wl_work_class_id - Workload work class identifier monitor element 1059
wlm_queue_assignments_total - Workload manager total queue assignments monitor element 1059
wlm_queue_time_total - Workload manager total queue time monitor element 1060
wlo_completed_total - Workload occurrences completed total monitor element 1062
work_action_set_id - Work action set ID monitor element 1062
work_action_set_name - Work action set name monitor element 1062
work_class_id - Work class ID monitor element 1063
work_class_name - Work class name monitor element 1063
workload_id - Workload ID monitor element . . 1063
workload_name - Workload name monitor element 1064
workload_occurrence_id - Workload occurrence identifier monitor element. 1065
workload_occurrence_state - Workload occurrence state monitor element 1066
x_lock_escals - Exclusive lock escalations monitor element 1066
xda_object_pages - XDA Object Pages 1067
xda_object_l_pages - XML storage object (XDA) data logical pages monitor element. 1068
xid - Transaction ID 1068
xquery_stmts - XQuery Statements Attempted 1068

Part 3. Appendixes 1071

Appendix A. Overview of the DB2 technical information 1073
DB2 technical library in hardcopy or PDF format 1073
Ordering printed DB2 books 1076
Displaying SQL state help from the command line processor 1077
Accessing different versions of the DB2 Information Center 1077
Displaying topics in your preferred language in the DB2 Information Center 1078
Updating the DB2 Information Center installed on your computer or intranet server 1078
Manually updating the DB2 Information Center installed on your computer or intranet server . . 1080
DB2 tutorials 1081
DB2 troubleshooting information 1082
Terms and Conditions 1082

Appendix B. Notices 1085

Index 1089

About this book

The *System Monitor Guide and Reference* describes how to collect different kinds of information about your database and the database manager.

It also explains how you can use the information you collected to understand database activity, improve performance, and determine the cause of problems.

© Copyright IBM Corp. 1993, 2012

Part 1. Monitoring interfaces

© Copyright IBM Corp. 1993, 2012

Chapter 1. Database monitoring

Database monitoring is a vital activity for the maintenance of the performance and health of your database management system. To facilitate monitoring, DB2® collects information from the database manager, its databases, and any connected applications. With this information you can do the following, and more:

- Forecast hardware requirements based on database usage patterns.
- Analyze the performance of individual applications or SQL queries.
- Track the usage of indexes and tables.
- Pinpoint the cause of poor system performance.
- Assess the impact of optimization activities (for instance, altering database manager configuration parameters, adding indexes, or modifying SQL queries).

© Copyright IBM Corp. 1993, 2012

Chapter 2. Monitor table functions overview

Starting with DB2 Version 9.7, you can access monitor data through a light-weight alternative to the traditional system monitor. Use monitor table functions to collect and view data for systems, activities, or data objects.

Data for monitored elements are continually accumulated in memory and available for querying. You can choose to receive data for a single object (for example, service class A or table TABLE1) or for all objects.

When using these table functions in a database partitioned environment, you can choose to receive data for a single partition or for all partitions. If you choose to receive data for all partitions, the table functions return one row for each partition. Using SQL, you can sum the values across partitions to obtain the value of a monitor element across partitions.

Monitoring system information using table functions

The system monitoring perspective encompasses the complete volume of work and effort expended by the data server to process application requests. From this perspective, you can determine what the data server is doing as a whole as well as for particular subsets of application requests.

Monitor elements for this perspective, referred to as request monitor elements, cover the entire range of data server operations associated with processing requests.

Request monitor elements are continually accumulated and aggregated in memory so they are immediately available for querying. Request monitor elements are aggregated across requests at various levels of the workload management (WLM) object hierarchy: by unit of work, by workload, by service class. They are also aggregated by connection.

Use the following table functions for accessing current system monitoring information:

- MON_GET_SERVICE_SUBCLASS and MON_GET_SERVICE_SUBCLASS_DETAILS
- MON_GET_WORKLOAD and MON_GET_WORKLOAD_DETAILS
- MON_GET_CONNECTION and MON_GET_CONNECTION_DETAILS
- MON_GET_UNIT_OF_WORK and MON_GET_UNIT_OF_WORK_DETAILS

This set of table functions enables you to drill down or focus on request monitor elements at a particular level of aggregation. Table functions are provided in pairs: one for relational access to commonly used data and the other for XML access to the complete set of available monitor elements.

The system monitoring information is collected by these table functions by default for a new database. You can change default settings using one or both of the following settings:

- The database configuration parameter **mon_req_metrics** specifies the minimum level of collection in all service classes.

© Copyright IBM Corp. 1993, 2012

- The COLLECT REQUEST METRICS clause of the CREATE/ALTER SERVICE CLASS statement specifies the level of collection for a service superclass. Use this setting to increase the level of collection for a given service class over the minimum level of collection set for all service classes.

The possible values for each setting are the following:

None No request monitor elements are collected

Base All request monitor elements are collected

For example, to collect system monitoring information for only a subset of service classes, do the following:

1. Set the database configuration parameter **mon_req_metrics** to NONE.
2. For each desired service class, set the COLLECT REQUEST METRICS clause of the CREATE/ALTER SERVICE CLASS statement to BASE.

Monitoring activities using table functions

The activity monitoring perspective focuses on the subset of data server processing related to executing activities. In the context of SQL statements, the term activity refers to the execution of the section for a SQL statement.

Monitor elements for this perspective, referred to as activity monitor elements, are a subset of the request monitor elements. Activity monitor elements measure aspects of work done for statement section execution. Activity monitoring includes other information such as SQL statement text for the activity.

For activities in progress, activity metrics are accumulated in memory. For activities that are SQL statements, activity metrics are also accumulated in the package cache. In the package cache activity metrics are aggregated over all executions of each SQL statement section.

Use the following table functions to access current data for activities:

MON_GET_ACTIVITY_DETAILS
Returns data about the individual activities in progress when the table function is called. Data is returned in a relational form, however, the detailed metrics are returned in an XML document in the DETAILS column of the results table.

MON_GET_PKG_CACHE_STMT
Returns a point-in-time view of both static and dynamic SQL statements in the database package cache. Data is returned in a relational form.

MON_GET_PKG_CACHE_STMT_DETAILS
Returns detailed metrics for one or more package cache entries. Data is returned in a relational form, however, the detailed metrics are returned in an XML document in the DETAILS column of the results table.

Activity monitoring information is collected by default for a new database. You can change default settings using one or both of the following settings:

- The **mon_act_metrics** database configuration parameter specifies the minimum level of collection in all workloads.
- The COLLECT ACTIVITY METRICS clause of the CREATE/ALTER WORKLOAD statement specifies the level of collection for a given workload over the minimum level of collection set for all workloads.

The possible values for each setting are the following:

None No activity monitor elements are collected

Base All activity monitor elements are collected

For example, to collect activity monitor elements for only selected workloads, do the following:

1. Set the **mon_act_metrics** database configuration parameter to NONE.
2. Set the COLLECT ACTIVITY METRICS clause of the CREATE/ALTER WORKLOAD statement to BASE. By default, the values for other workloads is NONE.

Monitoring data objects using table functions

The data object monitoring perspective provides information about operations performed on data objects, that is tables, indexes, buffer pools, table spaces, and containers.

A different set of monitor elements is available for each object type. Monitor elements for a data object are incremented each time a request involves processing that object. For example, when processing a request that involves reading rows from a particular table, the metric for rows read is incremented for that table.

Use the following table functions to access current details for data objects:

- MON_GET_BUFFERPOOL
- MON_GET_TABLESPACE
- MON_GET_CONTAINER
- MON_GET_TABLE
- MON_GET_INDEX

These table functions return data in a relational form.

You cannot access historical data for data objects.

Data object monitor elements are collected by default for new databases. You can use the **mon_obj_metrics** database configuration parameter to reduce the amount of data collected by the table functions.

The possible values for this configuration parameter are the following:

None No data object monitor elements are collected

Base All data object monitor elements are collected

Regardless of the what you set the **mon_obj_metrics** parameter to, data is always collected for monitor elements reported by the following table functions:

- MON_GET_TABLE
- MON_GET_INDEX

To stop collecting data object monitor elements reported by the following table functions, set the **mon_obj_metrics** configuration parameter to NONE.

- MON_GET_BUFFERPOOL
- MON_GET_TABLESPACE
- MON_GET_CONTAINER

Monitoring locking using table functions

You can retrieve information about locks using table functions. Unlike request, activity or data object monitor elements, information about locks is always available from the database manager. You do not need to enable the collection of this information.

Use the following monitor table functions to access current information for locks in the system:

- MON_GET_LOCKS
- MON_GET_APPL_LOCKWAIT

Both table functions return data in relational form.

Monitoring system memory using table functions

You can retrieve information about system memory usage using table functions.

You can examine memory usage at the level of memory sets, which are allocations of memory from the operating system. You can also examine memory usage by specific memory pools within a given memory set. Use the following monitor functions to access current information about memory usage:

- MON_GET_MEMORY_SET
- MON_GET_MEMORY_POOL

Other monitoring table functions

Besides table functions that return information about the system, activities, locks, or data objects there are also table functions that return various types of miscellaneous information These functions include ones that return information related to the fast communications manager (FCM), and about the status of table space extent movement.

Each of the table functions that follow can be used at any time. Unlike the table functions that return request metrics (the system monitoring perspective), activity metrics (the activity monitoring perspective) or metrics related to data objects (the data object monitoring perspective), it is not necessary to first enable the collection of the monitor elements returned by these functions.

- MON_GET_FCM
- MON_GET_FCM_CONNECTION_LIST
- MON_GET_EXTENT_MOVEMENT_STATUS

Chapter 3. Interfaces that return monitor data in XML documents

Starting in DB2 Version 9.7, some monitor data is reported as elements in XML documents.

Using XML to report monitor information provides improved extensibility and flexibility. New monitor elements can be added to the product without having to add new columns to an output table. Also, XML documents can be processed in a number of ways, depending on your needs. For example:

- You can use XQuery to run queries against the XML document.
- You can use the XSLTRANSFORM scalar function to transform the document into other formats.
- You can view their contents as formatted text using built-in MON_FORMAT_XML_* formatting functions, or the XMLTABLE table function.

XML documents containing monitor elements are produced by several monitoring interfaces. The sections that follow describe how results are returned as XML documents.

- "Monitor table functions with names that end with _DETAILS"
- "XML data returned by event monitors" on page 10.

Monitor table functions with names that end with "_DETAILS"

Examples of these table functions include:

- MON_GET_PKG_CACHE_STMT_DETAILS
- MON_GET_WORKLOAD_DETAILS
- MON_GET_CONNECTION_DETAILS
- MON_GET_SERVICE_SUBCLASS_DETAILS
- MON_GET_ACTIVITY_DETAILS
- MON_GET_UNIT_OF_WORK_DETAILS

These table functions return monitor elements from the system and the activity monitoring perspectives. Most of the monitor elements returned by these functions are contained in an XML document. For example, the MON_GET_CONNECTION_DETAILS table function returns the following columns:

- APPLICATION_HANDLE
- MEMBER
- DETAILS

The DETAILS column of each row contains an XML document that contains monitor element data. This XML document is composed of several document elements that correspond to monitor elements. Figure 1 on page 10 illustrates the DETAILS column containing the XML documents. In addition, it show monitor elements returned in the XML documents in the DETAILS column.

© Copyright IBM Corp. 1993, 2012

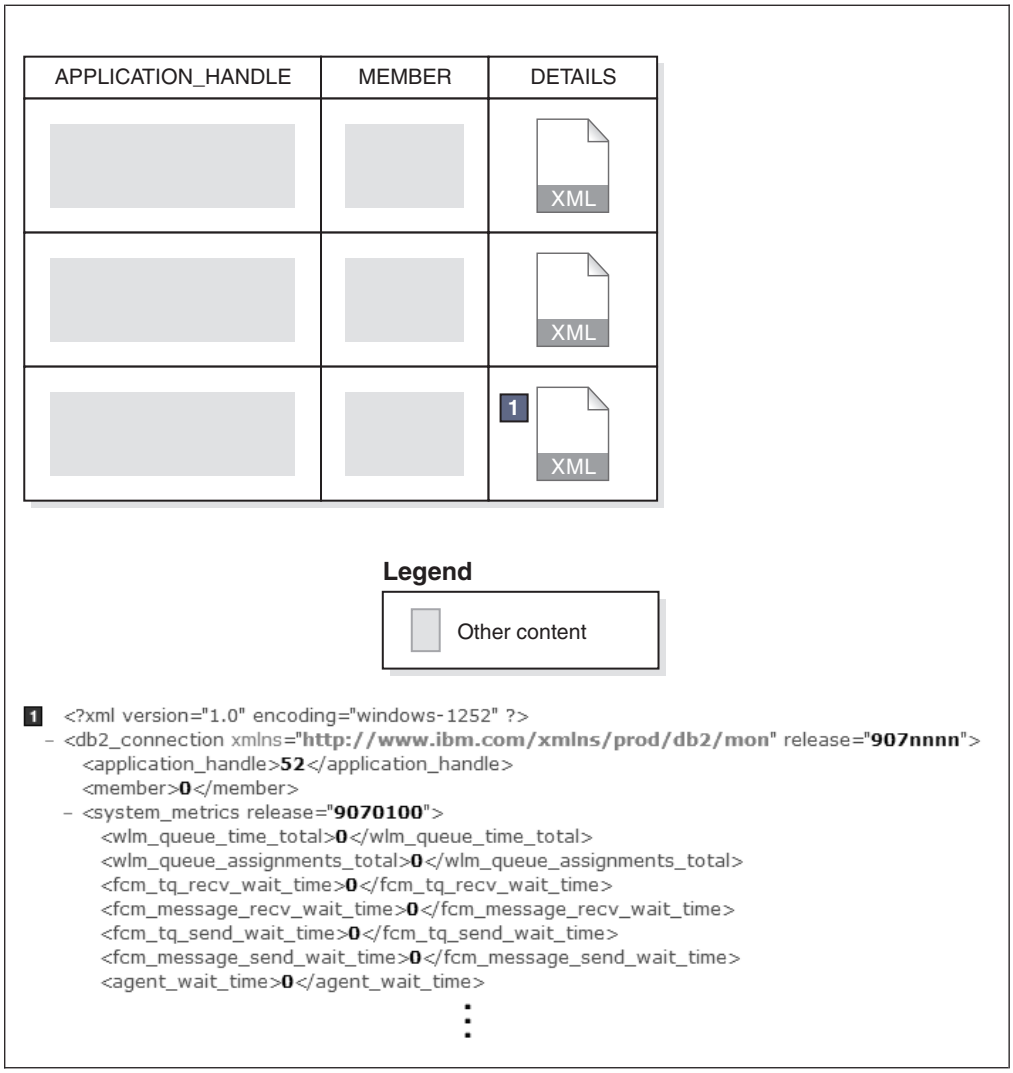


Figure 1. Table returned by MON_GET_CONNECTION_DETAILS, showing the DETAILS column containing XML documents. The contents of the XML document in the third row (**1**) are shown following the table.

In the preceding example, the <agent_wait_time> XML document element corresponds to **agent_wait_time** monitor element.

The schema for the XML document that is returned in the DETAILS column is available in the file `sqllib/misc/DB2MonRoutines.xsd`. Further details can be found in the file `sqllib/misc/DB2MonCommon.xsd`.

Some of the monitor elements contained in the document in the DETAILS column might be grouped into higher-level document elements. For example, monitor elements that report on activity-related metrics are part of the **activity_metrics** element. Similarly, system-level metrics are part of the **system_metrics** element.

XML data returned by event monitors

Several event monitors return data in XML format. They are summarized in Table 1 on page 11. Details about the XML documents returned by the various event monitor are described in the sections that follow.

Table 1. XML documents returned by various event monitors

Event monitor	Event monitor output format	XML document returned
"Statistics event monitor"	• Relational table • File • Named pipe	• metrics The metrics reported in this document are reset after every collection interval. • details_xml The metrics reported in this document accumulate until the database is deactivated. **Important:** Starting with Version 9.7 Fix Pack 6, the XML document details_xml is deprecated in the statistics event monitor, and might be removed in a future release. For more information, see Reporting of metrics in details_xml by the statistics event monitor has been deprecated"Reporting of metrics in details_xml by the statistics event monitor has been deprecated" in *What's New for DB2 Version 9.7*.
"Activity event monitor" on page 13	• Relational table • File • Named pipe	details_xml
"Package cache event monitor" on page 13	Unformatted event (UE) table	metrics This document can be viewed only after the UE table has been transformed to either XML or relational tables.
"Unit of work event monitor" on page 13	Unformatted event (UE) table	metrics This document can be viewed only after the UE table has been transformed to either XML or relational tables.

Statistics event monitor

The statistics event monitor records metrics in XML format when either of the twe following logical data groups are included in the event monitor output:

- event_scstats
- event_wlstats.

When you create a statistics event monitor to report on monitor elements in either of these groups, two XML documents are produced: `metrics` and `details_xml`. Both documents contain the same set of monitor elements that reflect metrics information. In the `metrics` document, the values of the elements show the change in value for the monitor element since the last time statistics were collected. The values of the elements contained in `details_xml` are not reset at each interval; they are reset only when the database is reactivated. If the data is written to a file or named pipe, these elements are part of the self-describing data stream. If the event monitor data is written to a table, the `metrics` document is stored in a column called METRICS; `details_xml` is stored in a column called DETAILS_XML. (Note: In these topics, when "metrics" or "details_xml" appears in lower-case letters, it refers to an XML document named metrics, or details_xml. METRICS, or

DETAILS_XML, in upper-letters, refers to a column in a relational table called METRICS or DETAILS_XML that contains the metrics or details_xml documents.)Figure 2 shows the XML documents in the METRICS and DETAILS_XML columns as they appear in the SCSTATS table produced by the statistics event monitor:

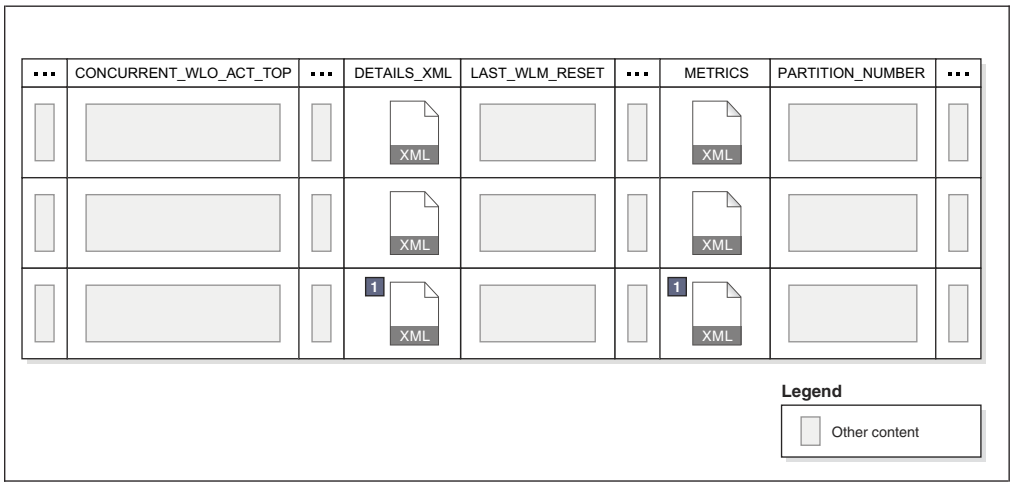


Figure 2. Output of statistics event monitor (when written to a table), showing the DETAILS_XML and METRICS columns.. The contents of the XML document in the third row (**1**) are shown following the table.

Each of these the document contained in these columns contains the **system_metrics** monitor element, which, in turn, contains a number of monitor elements that report on metrics related to system.

```
1  <?xml version="1.0" encoding="windows-1252" ?>
 - <db2_connection xmlns="http://www.ibm.com/xmlns/prod/db2/mon" release="907nnnn">
     <application_handle>52</application_handle>
     <member>0</member>
   - <system_metrics release="9070100">
       <wlm_queue_time_total>0</wlm_queue_time_total>
       <wlm_queue_assignments_total>0</wlm_queue_assignments_total>
       <fcm_tq_recv_wait_time>0</fcm_tq_recv_wait_time>
       <fcm_message_recv_wait_time>0</fcm_message_recv_wait_time>
       <fcm_tq_send_wait_time>0</fcm_tq_send_wait_time>
       <fcm_message_send_wait_time>0</fcm_message_send_wait_time>
       <agent_wait_time>0</agent_wait_time>
                                        :
```

Notes:

- Starting with Version 9.7 Fix Pack 6, the XML document details_xml is deprecated in the statistics event monitor, and might be removed in a future release. For more information, see Reporting of metrics in details_xml by the statistics event monitor has been deprecated"Reporting of metrics in details_xml by the statistics event monitor has been deprecated" in *What's New for DB2 Version 9.7*.
- The **system_metrics** element that is reported in the details_xml document contained in the DETAILS_XML column produced by the statistics event monitor is also a part of the XML document contained in the DETAILS column returned by the MON_GET_SERVICE_SUBCLASS_DETAILS and MON_GET_WORKLOAD_DETAILS table functions. Like the metrics reported in

the details_xml document, the values for the metrics reported in the document contained in the DETAILS column accumulate until the database is deactivated.

- See "Information written to XML for system_metrics and activity_metrics monitor elements" on page 129 for the schema for the XML output from a statistics event monitor.

Activity event monitor

When you create an activity event monitor to report on monitor elements in the event_activity logical data group (see "event_activity logical data group" on page 458), one of the columns produced is DETAILS_XML. If the event monitor is written to a table, DETAILS_XML is a column. If it is written to a file or named pipe, DETAILS_XML is part of the self-describing data stream. Either way, the document contains the **activity_metrics** monitor element, which, in turn, contains a number of monitor elements that report on metrics related to activities. See "Information written to XML for system_metrics and activity_metrics monitor elements" on page 129 for the schema for the XML output from an activity event monitor.

Note: activity_metrics as reported in the XML document in the DETAILS_XML column produced by the activity event monitor is also a part of the XML document contained in the DETAILS column returned by the MON_GET_ACTIVITY_DETAILS table function.

Package cache event monitor

The package cache event monitor writes its output to an unformatted event (UE) table. If you convert the data in this table using the EVMON_FORMAT_UE_TO_TABLES table function, one of the tables produced is PKGCACHE_EVENT. This table contains a METRICS column. In each row, this column contains an XML document with elements associated with package cache event monitor elements.

Note: Starting in DB2 Version 9.7 Fix Pack 1, EVMON_FORMAT_UE_TO_TABLES also creates a separate table for the metrics collected by this event monitor called PKGCACHE_METRICS. This table contains the same information reported in the METRICS column of the PKGCACHE_EVENT table. So, you can retrieve metrics from the columns of the PKGCACHE_METRICS table, or you can use the use the XML document contained in the METRICS column of the PKGCACHE_EVENT table. See "Information written to relational tables for a package cache event monitor" on page 87 for details.

The EVMON_FORMAT_UE_TO_XML function also produces an XML document with elements associated with package cache event monitor elements. For example, the XML document element <num_executions> corresponds to the **num_executions** monitor element. See "Information written to XML for a package cache event monitor" on page 81 for the schema for the XML output from a package cache event monitor.

Unit of work event monitor

The unit of work event monitor writes its output to an unformatted event (UE) table. If you convert the data in this table using the EVMON_FORMAT_UE_TO_TABLES table function, one of the tables produced is UOW_EVENT. This table contains a METRICS column, which contains an XML document with elements associated with unit of work event monitor elements.

Note: Starting in DB2 Version 9.7 Fix Pack 1, EVMON_FORMAT_UE_TO_TABLES also creates a separate table for the metrics collected by this event monitor called UOW_METRICS. This table contains the same information reported in the METRICS column of the UOW_EVENT table. So, you can retrieve metrics from the columns of the UOW_METRICS table, or you can use the use the XML document contained in the METRICS column of the UOW_EVENT table. See "Information written to relational tables for a unit of work event monitor" on page 119 for details.

The EVMON_FORMAT_UE_TO_XML function also produces an XML document with elements associated with unit of work event monitor elements. For example, the XML document element <workload_name> corresponds to the **workload_name** monitor element. See "Information written to XML for a unit of work event monitor" on page 107 for the schema for the XML output from a unit of work event monitor.

Interfaces for viewing XML monitor information as formatted text

You can view the data contained in the XML documents produced by monitor interfaces in several ways, depending on how you want to view or use the data. You can use XQuery to query and manipulate the XML documents returned by monitoring interfaces. You can also use table functions to format the XML documents for easier reading.

XQuery provides a powerful and flexible interface for querying and manipulating XML data. However, there are times where you might want to view element data in a text-based format. Depending on your needs, you can view monitor elements contained in an XML document in column- or row-oriented format. The former is useful if you know which monitor elements you want to see. The latter is useful if you do not know ahead of time which monitor elements you want to examine, such as when you want to see the top five types of wait times. The sections that follow describe two ways that you can view monitor data contained in XML documents as formatted text.

- "Viewing monitor elements in column-oriented format"
- "Viewing monitor elements in row-oriented format" on page 15

Viewing monitor elements in column-oriented format

The XMLTABLE table function takes an XML document as input and coverts it into a relational table such that each of the selected XML document elements appears as a column. This approach is useful if you know which monitor elements you want to display. For example, assume that you have created a statistics event monitor called DBSTATS to collect information from the event_scstats logical data group. (See "event_scstats logical data group" on page 478 for more information about the monitor elements associated with this logical data group.) The monitor elements in this logical group include **metrics**, which is an XML document that itself contains the metrics that make up the system_metrics monitor element. (See "system_metrics" on page 129 for more information about the monitor elements associated with the system_metrics monitor element.) To view specific system_metrics monitor elements contained in **metrics**, such as rows_returned, total_section_time, or total_cpu_time, you can use the XMLTABLE table function to

1. Note: In these topics, when "metrics" or "details_xml" appears in lower-case letters, it refers to an XML document named **metrics**, or **details_xml**. METRICS, or DETAILS_XML, in upper-letters, refers to a column in a relational table called METRICS or DETAILS_XML that contains the `metrics` or `details_xml` documents.

format selected monitor elements from the **metrics** documents returned by the statistics event monitor. The example that follows illustrates this. (For presentation purposes, the SQL returns results only for a specific service class.)

```
SELECT partition_number,
       service_class_id,
       statistics_timestamp,
       event.rows_returned,
       event.total_section_time,
       event.total_cpu_time
 FROM  SCMETRICS_DBSTATS as DBSTATS,
 XMLTABLE( XMLNAMESPACES( DEFAULT 'http://www.ibm.com/xmlns/prod/db2/mon' ),
           '$metrics/system_metrics' PASSING XMLPARSE( DOCUMENT DBSTATS.METRICS ) as "metrics"
           COLUMNS
           rows_returned         BIGINT      PATH 'rows_returned',
           total_section_time    BIGINT      PATH 'total_section_time',
           total_cpu_time        BIGINT      PATH 'total_cpu_time'
         ) AS EVENT
 WHERE service_class_id = 12;
```

The following output shows the results for this query:

```
PARTITION_NUMBER SERVICE_CLASS_ID STATISTICS_TIMESTAMP       ROWS_RETURNED    TOTAL_SECTION_TIME TOTAL_CPU_TIME
---------------- --------------- -------------------------- ---------------- ------------------ ----------------
               0              12 2010-01-05-12.14.37.001717              402                990          1531250
               0              12 2010-01-05-12.15.00.035409              402                990          1531250
               0              12 2010-01-05-12.20.00.021884              412               1064          1609375
               0              12 2010-01-05-12.25.00.039175              422               1075          1687500
               0              12 2010-01-05-12.29.59.950137              432               1104          1765625
               0              12 2010-01-05-12.34.59.948979              442               1130          1796875
               0              12 2010-01-05-12.39.59.903928              452               1149          1890625
               0              12 2010-01-05-12.44.59.953596              462               1178          1953125
               0              12 2010-01-05-12.49.59.970059              473               1207          2062500
               0              12 2010-01-05-12.54.59.971990              483               1230          2109375

  10 record(s) selected.
```

In this case, the first three columns are displayed directly from the table SCMETRICS_DBSTATS table produced by the statistics event monitor. The last three columns are metrics monitor elements extracted from the XML document in the METRICS column of the table.

For more information about using XMLTABLE, refer to the documentation for that function. You can also see examples of using XMLTABLE to view monitor elements in the documentation for the various MON_GET_*_DETAILS functions.

Viewing monitor elements in row-oriented format

The table functions with names of the form MON_FORMAT_XML_* _BY_ROW introduced in DB2 Version 9.7 Fix Pack 1 provide a quick way to display the metrics monitor elements contained in an XML document. They report metrics in a row-based format, with each monitor element appearing in a row by itself. The following functions are included in this group:

- MON_FORMAT_XML_COMPONENT_TIMES_BY_ROW
- MON_FORMAT_XML_TIMES_BY_ROW
- MON_FORMAT_XML_WAIT_TIMES_BY_ROW
- MON_FORMAT_XML_METRICS_BY_ROW

For example, the XML document returned by the statistics event monitor, DETAILS_XML, might look something like the one shown in the first part of Figure 3 on page 16. If you use the MON_FORMAT_XML_WAIT_TIMES_BY_ROW

function to format the content of DETAILS_XML, the output would look like the table at the bottom of the diagram.

```
<?xml version="1.0" encoding="windows-1252" ?>
- <system_metrics xmlns="http://www.ibm.com/xmlns/prod/db2/mon" release="907nnnn">
    <wlm_queue_time_total>0</wlm_queue_time_total>
    <wlm_queue_assignments_total>0</wlm_queue_assignments_total>
    <fcm_tq_recv_wait_time>0</fcm_tq_recv_wait_time>
    <fcm_message_recv_wait_time>0</fcm_message_recv_wait_time>
    <fcm_tq_send_wait_time>0</fcm_tq_send_wait_time>
    <fcm_message_send_wait_time>0</fcm_message_send_wait_time>
    <agent_wait_time>0</agent_wait_time>
    <agent_waits_total>0</agent_waits_total>
    <lock_wait_time>0</lock_wait_time>
```

⋮

MON_FORMAT_XML_WAIT_TIMES_BY_ROW(DETAILS_XML)

```
METRIC_NAME                       TOTAL_TIME_VALUE     COUNT                PARENT_METRIC_NAME
--------------------------------- -------------------- -------------------- ----------------------------
WLM_QUEUE_TIME_TOTAL                                 0                    0 TOTAL_WAIT_TIME
FCM_TQ_RECV_WAIT_TIME                                0                    0 FCM_RECV_WAIT_TIME
FCM_MESSAGE_RECV_WAIT_TIME                           0                    0 FCM_RECV_WAIT_TIME
FCM_TQ_SEND_WAIT_TIME                                0                    0 FCM_SEND_WAIT_TIME
FCM_MESSAGE_SEND_WAIT_TIME                           0                    0 FCM_SEND_WAIT_TIME
AGENT_WAIT_TIME                                      0                    0 TOTAL_WAIT_TIME
LOCK_WAIT_TIME                                       0                    0 TOTAL_WAIT_TIME
DIRECT_READ_TIME                                     0                    0 TOTAL_WAIT_TIME
DIRECT_WRITE_TIME                                    0                    0 TOTAL_WAIT_TIME
LOG_BUFFER_WAIT_TIME                                 0                    0 TOTAL_WAIT_TIME
LOG_DISK_WAIT_TIME                                   0                    0 TOTAL_WAIT_TIME
```

⋮

Figure 3. An XML file containing monitoring data, processed by one of the MON_FORMAT_XML_* functions. This example shows the use of the MON_FORMAT_XML_WAIT_TIMES_BY_ROW function. Only wait times are returned; other metrics contained in the XML file, such as wlm_queue_assignments_total are excluded by this particular function.

The number of columns returned varies by the specific function that you use. For example MON_FORMAT_XML_METRICS_BY_ROW returns two columns, one for the metric name, and one for its corresponding value:

```
METRIC_NAME               VALUE
------------------------- --------------------
WLM_QUEUE_TIME_TOTAL                         0
WLM_QUEUE_ASSIGNMENTS_TOT                    0
FCM_TQ_RECV_WAIT_TIME                        0
FCM_MESSAGE_RECV_WAIT_TIM                    0
FCM_TQ_SEND_WAIT_TIME                        0
```

⋮

By comparison, MON_FORMAT_XML_TIMES_BY_ROW returns four columns:

```
METRIC_NAME                   TOTAL_TIME_VALUE     COUNT      PARENT_METRIC_NAME
----------------------------- -------------------- ---------- ----------------------
WLM_QUEUE_TIME_TOTAL                           0          0 TOTAL_WAIT_TIME
FCM_TQ_RECV_WAIT_TIME                          0          0 FCM_RECV_WAIT_TIME
FCM_MESSAGE_RECV_WAIT_TIME                     0          0 FCM_RECV_WAIT_TIME
FCM_TQ_SEND_WAIT_TIME                          0          0 FCM_SEND_WAIT_TIME
FCM_MESSAGE_SEND_WAIT_TIME                     0          0 FCM_SEND_WAIT_TIME
```

⋮

The MON_FORMAT_XML_* _BY_ROW functions are useful when you do not know which elements you want to view. For example, you might want to see the top 10 wait-time monitor elements for the workload named CLPWORKLOAD. To collect this information, you can create a statistics event monitor called DBSTATS (event_wlstats logical data group). Assuming you set up this event monitor to write to a table, it records metrics in a column called DETAILS_XML. Once the output table from the event monitor is populated with monitor data, you can construct a query that uses the MON_FORMAT_XML_WAIT_TIMES_BY_ROW function to extract the monitor elements you want to see:

```
SELECT SUBSTR(STATS.WORKLOAD_NAME,1,15) AS WORKLOAD_NAME,
       SUBSTR(METRICS.METRIC_NAME,1,30) AS METRIC_NAME,
       SUM(METRICS.TOTAL_TIME_VALUE) AS TOTAL_TIME_VALUE
FROM   WLSTATS_DBSTATS AS STATS,
       TABLE(MON_FORMAT_XML_WAIT_TIMES_BY_ROW(STATS.DETAILS_XML)) AS METRICS
WHERE  WORKLOAD_NAME='CLPWORKLOAD' AND (PARENT_METRIC_NAME='TOTAL_WAIT_TIME')
GROUP  BY WORKLOAD_NAME,METRIC_NAME
ORDER  BY TOTAL_TIME_VALUE DESC
FETCH FIRST 10 ROWS ONLY
```

Remember: Time spent monitor elements are organized into hierarchies. In this example, to avoid double-counting wait times, only the monitor elements that roll-up to **total_wait_time** are included (see the WHERE clause in the preceding SQL statement). Otherwise, total_wait_time itself would be included in the results, which includes several individual wait times.
The output that follows shows what the results of the preceding query might look like:

```
WORKLOAD_NAME   METRIC_NAME                    TOTAL_TIME_VALUE
--------------- ------------------------------ --------------------
CLPWORKLOAD     LOCK_WAIT_TIME                             15138541
CLPWORKLOAD     DIRECT_READ_TIME                            6116231
CLPWORKLOAD     POOL_READ_TIME                              6079458
CLPWORKLOAD     DIRECT_WRITE_TIME                            452627
CLPWORKLOAD     POOL_WRITE_TIME                              386208
CLPWORKLOAD     IPC_SEND_WAIT_TIME                           283172
CLPWORKLOAD     LOG_DISK_WAIT_TIME                           103888
CLPWORKLOAD     DIAGLOG_WRITE_WAIT_TIME                       78198
CLPWORKLOAD     IPC_RECV_WAIT_TIME                            15612
CLPWORKLOAD     TCPIP_SEND_WAIT_TIME                           3291

  10 record(s) selected.
```

Note: The MON_FORMAT_XML_*_BY_ROW functions return only monitor elements that track measurements or *metrics*. These include monitor elements that track wait and component times, as well as counters. They do not return non-metrics monitor elements contained in the XML document, such as uow_id, or activity_id.

You can use the XMLTABLE function to view *any* of the elements (including non-metrics elements) contained in the XML document. However, the most frequently used, non-metrics monitor elements are returned as columns by the monitor functions that begin with MON_GET_*, such as MON_GET_UNIT_OF_WORK, or MON_GET_CONNECTION. If you are not familiar with XML, you might find it faster and easier to create queries using these functions than using the XMLTABLE function to extract monitor elements from an XML document.

To summarize: if you are interested in viewing non-metrics monitor elements, the MON_GET_* series of table functions might be a good alternative to the XMLTABLE function. If you are interested in viewing metrics monitor elements, the MON_FORMAT_XML_*_BY_ROW table functions might suit your needs.

Viewing metrics monitor elements from XML documents as rows in a table

One way to view metrics-related information contained in an XML document returned from an event monitor is to convert it into a format where each monitor element is displayed in a row by itself. This format is useful if you want to view the information in a text-based format, but do not know specifically which monitor elements you want to examine.

About this task

To view metrics information in row-based format from the XML documents returned by various monitoring interfaces, use the MON_FORMAT_XML_* _BY ROW table functions. These functions were introduced in DB2 Version 9.7 Fix Pack 1.

Procedure

The example shown in this task uses the MON_FORMAT_XML_TIMES_BY_ROW table function to view component times for a statement as tracked by the package cache event monitor. It assumes that a package cache event monitor called PKGCACHEEVENTS has been created and activated. The package cache event monitor writes its output to an unformatted event (UE) table. Before it can be used, the data in the UE table must be converted to either relational tables using the EVMON_FORMAT_UE_TO_TABLES stored procedure, or to XML using the EVMON_FORMAT_UE_TO_XML table function. This task shows the first of these two approaches.

1. First, convert the unformatted event (UE) table that the package cache event monitor writes to into relational tables using the EVMON_FORMAT_UE_TO_TABLES procedure:

```
call EVMON_FORMAT_UE_TO_TABLES ('PkgCache',NULL,NULL,NULL,NULL,NULL,NULL,0,'SELECT * FROM PKGCACHEEVENTS')
```

   This procedure creates two tables:

   - One is called PKGCACHE_EVENT, which contains a column called METRICS. This column, in turn, contains XML documents with metrics monitor elements.
   - The other is called PKGCACHE_METRICS.

     Note: You could view the metrics directly from the columns in PKGCACHE_METRICS, rather than extract metrics from the METRICS column of the PKGCACHE_EVENT table. However, when you examine PKGCACHE_METRICS, the metrics are displayed in columns, rather than rows; it is not as easy to get a ranking of, say, the metrics with the highest values.

2. Query the two tables produced in the preceding step to determine which statement is the most expensive in terms of execution times:

```
SELECT EVENTS.EXECUTABLE_ID,
       SUM(METRICS.STMT_EXEC_TIME) AS TOTAL_STMT_EXEC_TIME
FROM   PKGCACHE_EVENT AS EVENTS,
       PKGCACHE_METRICS AS METRICS
WHERE  EVENTS.XMLID = METRICS.XMLID
GROUP  BY EVENTS.EXECUTABLE_ID
ORDER  BY TOTAL_STMT_EXEC_TIME DESC
FETCH  FIRST 5 ROWS ONLY
```

In the preceding query, the two tables produced in step 1 on page 18 are joined so that the statement IDs from the PKGCACHE_EVENT table can be associated with their execution times in the PKGCACHE_METRICS table:

```
EXECUTABLE_ID                                                                    TOTAL_STMT_EXEC_TIME
-------------------------------------------------------------------------------- --------------------
x'0100000000000001A0300000000000000000000020020091215115933859000'                  250
x'0100000000000000150300000000000000000000020020091215115850328000'                 191
x'0100000000000000210200000000000000000000020020091215115818343001'                 129
x'010000000000000C4020000000000000000000002002009121511583857800'                    41
x'010000000000000B0020000000000000000000002002009121511583820300'                    38

  5 record(s) selected.
```

The first item in the results represents the statement with the largest overall execution time.

3. Optional: If you like, you can display the text for the statement using the following SQL:

```
SELECT SUBSTR(STMT_TEXT,1,60) AS STMT_TEXT
FROM PKGCACHE_EVENT
WHERE EXECUTABLE_ID = x'0100000000000001A0300000000000000000000020020091215115933859000'
```

Results:

```
STMT_TEXT
------------------------------------------------------------
DROP XSROBJECT MYSCHEMA.EVMON_PKGCACHE_SCHEMA_SQL09070

  1 record(s) selected.
```

4. Use the MON_FORMAT_XML_TIMES_BY_ROW table function to view a listing of the time-spent monitor elements for the statement you identified in step 2 on page 18:

```
SELECT SUBSTR(XMLMETRICS.METRIC_NAME,1,30) AS METRIC_NAME,
       XMLMETRICS.TOTAL_TIME_VALUE,
       SUBSTR(XMLMETRICS.PARENT_METRIC_NAME,1,30) AS PARENT_METRIC_NAME
FROM PKGCACHE_EVENT AS EVENTS,
     TABLE(MON_FORMAT_XML_TIMES_BY_ROW(EVENTS.METRICS)) AS XMLMETRICS
WHERE
 EVENTS.EXECUTABLE_ID=x'0100000000000001A0300000000000000000000020020091215115933859000'
 AND PARENT_METRIC_NAME='STMT_EXEC_TIME'
ORDER BY XMLMETRICS.TOTAL_TIME_VALUE DESC
```

Notes:

- Remember that time-spent monitor elements are organized into hierarchies. To eliminate double-counting, only those metrics that roll-up to **stmt_exec_time** are included in the results. Otherwise, stmt_exec_time itself would be included in the results, which include several individual component times.
- PARENT_METRIC_NAME, one of the columns returned by MON_FORMAT_XML_TIMES_BY_ROW is included for illustrative purposes.

When run, the following results are returned by this query:

```
METRIC_NAME                  TOTAL_TIME_VALUE     PARENT_METRIC_NAME
---------------------------- -------------------- ----------------------------
TOTAL_ACT_WAIT_TIME                           234 STMT_EXEC_TIME
TOTAL_SECTION_PROC_TIME                        15 STMT_EXEC_TIME
```

Here, you can see that the total processing time adds up to 249 ms. Compare this time to the total time of 250 shown in step 2 on page 18; the extra millisecond is accounted for by other times (for example, waits) not included in **stmt_exec_time**.

Results

In the results from the preceding example, you can see the arrangement of the metrics: they are displayed in row-oriented format, one metric per row. The advantage of using this approach is that you do not need to know ahead of time which metrics or monitor elements you want to see. If you are interested in the top or bottom *n* metrics, or the metrics that fall within a specific range of values, you can easily create a query to return the results you are interested in. By contrast, if you use the XMLTABLE function to display the monitor elements as columns, you need to specify which monitor elements to display (or display them all).

Example

Viewing the contents of the DETAILS column produced by a MON_GET_*_DETAILS table function

You can also use the MON_FORMAT_XML_*_BY_ROW functions to view the contents of the DETAILS column returned by any of the MON_GET_*_DETAILS functions. For example, MON_GET_CONNECTION_DETAILS returns a DETAILS column that contains an XML document with metrics that pertain to a database connection.

For example, to view the non-zero component times for each connection across all members, you could use the following query:

```
SELECT CONDETAILS.APPLICATION_HANDLE,
       SUBSTR(XMLMETRICS.METRIC_NAME,1,30) AS METRIC_NAME,
       SUM(XMLMETRICS.TOTAL_TIME_VALUE) AS TOTAL_TIME_VALUE,
       SUBSTR(XMLMETRICS.PARENT_METRIC_NAME,1,30) AS PARENT_METRIC_NAME
FROM TABLE(MON_GET_CONNECTION_DETAILS(NULL,-1)) AS CONDETAILS,
     TABLE(MON_FORMAT_XML_COMPONENT_TIMES_BY_ROW(CONDETAILS.DETAILS))AS XMLMETRICS
WHERE TOTAL_TIME_VALUE > 0 AND XMLMETRICS.PARENT_METRIC_NAME='TOTAL_RQST_TIME'
GROUP BY CONDETAILS.APPLICATION_HANDLE,
         XMLMETRICS.PARENT_METRIC_NAME,
         XMLMETRICS.METRIC_NAME
ORDER BY CONDETAILS.APPLICATION_HANDLE ASC, TOTAL_TIME_VALUE DESC
```

Notes:

- To eliminate double-counting, only those metrics that roll-up to **total_rqst_time** are included in the results (`WHERE .... XMLMETRICS.PARENT_METRIC_NAME='TOTAL_RQST_TIME'`). Otherwise, total_rqst_time itself would be included in the results, which include several individual component times.
- PARENT_METRIC_NAME, one of the columns returned by MON_FORMAT_XML_COMPONENT_TIMES_BY_ROW is included for illustrative purposes.

The preceding query returns the following results:

```
APPLICATION_HANDLE   METRIC_NAME                   TOTAL_TIME_VALUE     PARENT_METRIC_NAME
-------------------- ------------------------------ -------------------- ------------------------------
                  52 TOTAL_SECTION_TIME                             3936 TOTAL_RQST_TIME
                  52 TOTAL_COMPILE_TIME                              482 TOTAL_RQST_TIME
                  52 TOTAL_COMMIT_TIME                                15 TOTAL_RQST_TIME
                  52 TOTAL_ROLLBACK_TIME                               1 TOTAL_RQST_TIME
                 496 TOTAL_COMPILE_TIME                              251 TOTAL_RQST_TIME
                 496 TOTAL_SECTION_TIME                               46 TOTAL_RQST_TIME
                 496 TOTAL_IMPLICIT_COMPILE_TIME                      5 TOTAL_RQST_TIME

  7 record(s) selected.
```

As this example shows, only metrics that comprise **total_rqst_time** are included. Had the `WHERE ....`

XMLMETRICS.PARENT_METRIC_NAME='TOTAL_RQST_TIME' clause not been included in the query, the results would look like those that follow:

```
APPLICATION_HANDLE   METRIC_NAME                   TOTAL_TIME_VALUE     PARENT_METRIC_NAME
-------------------- ---------------------------- -------------------- ----------------------------
                  52 TOTAL_RQST_TIME                                4603 -
                  52 TOTAL_SECTION_TIME                             3942 TOTAL_RQST_TIME
                  52 TOTAL_COMPILE_TIME                              537 TOTAL_RQST_TIME
                  52 TOTAL_SECTION_SORT_TIME                         299 TOTAL_SECTION_TIME
                  52 TOTAL_COMMIT_TIME                                15 TOTAL_RQST_TIME
                  52 TOTAL_ROLLBACK_TIME                               1 TOTAL_RQST_TIME
                 496 TOTAL_RQST_TIME                                 341 -
                 496 TOTAL_COMPILE_TIME                              251 TOTAL_RQST_TIME
                 496 TOTAL_SECTION_TIME                               46 TOTAL_RQST_TIME
                 496 TOTAL_IMPLICIT_COMPILE_TIME                      5 TOTAL_RQST_TIME
                 496 TOTAL_SECTION_SORT_TIME                          2 TOTAL_SECTION_TIME

  11 record(s) selected.
```

In this case, the values for **total_rqst_time** for each connection are included in the results, which includes the values for all other elements for which it is the parent. Similarly, the values for items in *italics* roll up to the **total_section_time**. Had they not been excluded in the WHERE clause, they would have been triple-counted in the results, as total_section_time itself rolls up to total_rqst_time.

Chapter 4. Event monitors

Event monitors return information for the event types specified in the CREATE EVENT MONITOR statement. For each event type, monitoring information is collected at a certain point in time.

The following table lists available event types, when the monitoring data is collected, and the information available for each event type. The available event types in the first column correspond to the keywords used in the CREATE EVENT MONITOR statement, where the event type is defined.

In addition to the defined events where data occurs, you can use the FLUSH EVENT MONITOR SQL statement to generate events. The events generated by this method are written with the current database monitor values for all the monitor types (except for DEADLOCKS and DEADLOCKS WITH DETAILS) associated with the flushed event monitor.

When monitoring the execution of SQL procedures using statement event monitors:

- Data manipulation language (DML) statements, such as INSERT, SELECT, DELETE, and UPDATE, generate events.
- Procedural statements, such as variable assignments and control structures (for example, WHILE or IF), do not generate events in a deterministic fashion.

Table 2. Event Types

Event type	When data is collected	Available information
DEADLOCKS[1]	Detection of a deadlock	Applications involved, and locks in contention.
DEADLOCKS WITH DETAILS[1]	Detection of a deadlock	Comprehensive information regarding applications involved, including the identification of participating statements (and statement text) and a list of locks being held. Using a DEADLOCKS WITH DETAILS event monitor instead of a DEADLOCKS event monitor will incur a performance cost when deadlocks occur, due to the extra information that is collected.
DEADLOCKS WITH DETAILS HISTORY[1]	Detection of a deadlock	All information reported in a DEADLOCKS WITH DETAILS event monitor, along with the statement history for the current unit of work of each application owning a lock participating in a deadlock scenario for the database partition where that lock is held. Using a DEADLOCKS WITH DETAILS HISTORY event monitor will incur a minor performance cost when activated due to statement history tracking.
DEADLOCKS WITH DETAILS HISTORY VALUES[1]	Detection of a deadlock	All information reported in a deadlock with details and history, along with the values provided for any parameter markers at the time of execution of a statement. Using a DEADLOCKS WITH DETAILS HISTORY VALUES event monitor will incur a more significant performance cost when activated due to extra copying of data values.

© Copyright IBM Corp. 1993, 2012

Table 2. Event Types (continued)

Event type	When data is collected	Available information
STATEMENTS	End of SQL statement	Statement start or stop time, CPU used, text of dynamic SQL, SQLCA (return code of SQL statement), and other metrics such as fetch count. **Note:** Statement start or stop time is unavailable when the Timestamp switch is off.
	End of subsection	For partitioned databases: CPU consumed, execution time, table and table queue information.
TRANSACTIONS[2]	End of unit of work	UOW work start or stop time, previous UOW time, CPU consumed, locking and logging metrics. Transaction records are not generated if running with XA.
CONNECTIONS	End of connection	All application level counters.
DATABASE	Database deactivation	All database level counters.
BUFFERPOOLS	Database deactivation	Counters for buffer pool, prefetchers, page cleaners and direct I/O for each buffer pool.
TABLESPACES	Database deactivation	Counters for buffer pool, prefetchers, page cleaners and direct I/O for each table space.
TABLES	Database deactivation	Rows read or written for each table.
Activities	Completion of an activity that executed in a service class, workload or work class that had its COLLECT ACTIVITY DATA option turned on. Data is also collected for the targeted activity at the instant the WLM_CAPTURE_ACTIVITY_IN_PROGRESS stored procedure is executed. Data is also collected if the activity violates a threshold that has the COLLECT ACTIVITY DATA option enabled.	Activity level data. If WITH DETAILS was specified as part of COLLECT ACTIVITY DATA, this will include statement and compilation environment information for those activities that have it. If AND VALUES was also specified, this will also include input data values for those activities that have it.
Statistics	Every *period* minutes, where *period* is the length of time over which statistics are gathered. This period is defined in the WLM_COLLECT_INT database configuration parameter. Data is also collected when the WLM_COLLECT_STATS stored procedure is called.	Statistics computed from the activities that executed within each service class, workload, or work class that exists on the system.
Threshold violations	Upon detection of a threshold violation.	Threshold violation information.
Locking	Upon detection of any of the following event types, depending on configuration settings : lock timeout, deadlock, lock wait beyond a specified duration.	Lock event records.
Unit of work	Upon completion of a unit of work	Unit of work event records. Option to include request metrics in the record.

[1] This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

[2] This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR UNIT OF WORK statement to monitor transaction events.

Note: A detailed deadlock event monitor is created for each newly created database. This event monitor, named DB2DETAILDEADLOCK, starts when the database is activated and will write to files in the database directory. You can avoid

the overhead this event monitor incurs by dropping it. The DB2DETAILDEADLOCK event monitor is deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Event monitors that write to an unformatted event table

DB2 9.7 introduces some event monitors with a new type of target, the unformatted event table. This type of event monitor provides better performance, new CREATE EVENT MONITOR statement options, and the new interfaces to access data for analysis.

Characteristics of unformatted event table event monitors affect how you can perform the following tasks:

- Creating the event monitor and configuring data collection
- Managing event monitor operations
- Accessing event data captured by the event monitor

Typically, you can achieve all your monitoring needs by creating a single event monitor per database for a given type of event (for example, lock events). You can alter settings to increase or decrease the amount of data that you can collect with the monitor to address changing monitoring needs. This contrasts with some older event monitors where a more common practice is to create a multiple event monitors, each geared to collect a particular monitoring need.

Creating the unformatted event table associated with an event monitor

One aspect of creating an event monitor is specifying where to write the data that the monitor collects. This type of event monitor always writes data in binary format to an unformatted event table. The unformatted event table is a target type introduced in DB2 9.7. An unformatted event table is created implicitly each time you create an event monitor. The CREATE EVENT statement for this type of event monitor includes the clause WRITE TO UNFORMATTED EVENT TABLE.

The CREATE EVENT MONITOR statement includes the following options for configuring the unformatted event table:

- table name - By default, the unformatted event table is named based on the event monitor name.
- tablespace name - By default, the unformatted table is created in the table space IBMDEFAULTGROUP over which the user has USE privilege if it exists. However, the recommended practice is to define a tablespace optimized for your event monitors, as described below.
- PCTDEACTIVATE - The default value is 100, which means that the event monitor deactivates when the table space becomes full.

The following considerations about the table space for the unformatted event table must be taken:

- Create a table space for your event monitor unformatted event tables that is configured for performance. Use the following clauses with the CREATE TABLESPACE statement:
  - Specify a page size (PAGESIZE) as large as possible, up to and including 32KB.
  - Specify the NO FILE CACHING SYSTEM option.

- In a partitioned database environment, consider on which partitions the table space exists. If a table space for a target unformatted event table does not exist on some database partition, data for that target unformatted event table is ignored. This behavior allows users to choose a subset of database partitions for monitoring to be chosen, by creating a table space that exists only on certain database partitions.

Other useful information about unformatted event tables includes the following:

- The SYSCAT.EVENTTABLES catalog view lists event monitors, their associated unformatted table, and other details.
- The columns of the unformatted event table are described in a topic listed in the related links.

Configuring data collection for an event monitor

Setting up an event monitor involves specifying what data to collect. Aspects include which subset of the system workload to monitor, what type of events to collect, how much detail to collect for each event, and enabling/disabling data capture (turning the data capture on and off). Considerations for configuring data collection are the following:

- With this type of event monitor, you configure data collection primarily by setting properties of individual workload definitions using the CREATE/ALTER WORKLOAD statement. That is, you can specify different data collection settings for different workloads. The CREATE/ALTER WORKLOAD statement includes clauses specific to particular types of event monitor.
- By default, this type of event monitor is automatically activated. You can specify that the event monitor be activated manually by specifying the MANUALSTART keyword in the CREATE EVENT MONITOR statement. You can then control the event monitor with the SET EVENT MONITOR STATE statement.
- As mentioned in another context, in a partitioned database environment you can choose a subset of database partitions to monitor with your event monitor. When you create the event monitor, specify a table space for the unformatted event table that resides only on those partitions you want to monitor. If the unformatted event table does not exist on a given database partition, the event monitor will not collect data for that partition.
- Data collection for this type of event monitor is not affected by system monitor switch settings set using the UPDATE MONITOR SWITCHES statement nor is event capture is turned on and off using the SET EVENT MONITOR STATE statement.

Managing event monitor operations

The following points provide guidance for managing ongoing operation of an event monitor:

- At any time, you can change your specification of what data to collect by using the ALTER WORKLOAD statement.
- If you specified the MANUALSTART option in the CREATE EVENT MONITOR statement, you can start and stop data collection using the SET EVENT MONITOR STATE statement.
- Unformatted event tables must be manually pruned.
- If an unformatted event table reaches the maximum space allotted, the event monitor will deactivate.

- If an event monitor is no longer needed, use the DROP statement to drop an event monitor. Issuing the DROP statement does not drop the unformatted event table that is associated with the event monitor. The associated unformatted event table must be manually dropped after the event monitor is dropped. If you don't drop the unformatted event table, you will encounter difficulties if you subsequently try to create another event monitor whose unformatted event table has the same name as an existing one.

Accessing event data captured by an event monitor

This type of event monitor writes data in a binary format to an unformatted event table. You can access this data using the db2evmonfmt command or routines provided for this purpose.

With the db2evmonfmt command you can:

- select events of interest based on the following attributes: event ID, event type, time period, application, workload, or service class.
- choose whether to receive the output in the form of a text report or a formatted XML document.
- completely control the output format by creating your own XSLT style sheets instead of using the ones provided with db2evmonfmt.

You can also extract data from an unformatted event table using the following routines:

- EVMON_FORMAT_UE_TO_XML - extracts data from an unformatted event table into an XML document.
- EVMON_FORMAT_UE_TO_TABLES - extracts data from an unformatted event table into a set of relational tables.

With these routines, you can use a SELECT statement to specify the exact rows from the unformatted event table that you want to extract.

Unformatted event table column definitions

An unformatted event table is created when you issue a CREATE EVENT MONITOR statement that includes the clause WRITE TO UNFORMATTED EVENT TABLE. The column definitions are useful when you want to extract data to analyze or prune a table of unneeded data.

The column definitions for the unformatted event table are useful when you want to extract data from an unformatted event table using one of the following routines:

- EVMON_FORMAT_UE_TO_XML - extracts data from an unformatted event table into an XML document.
- EVMON_FORMAT_UE_TO_TABLES - extracts data from an unformatted event table into a set of relational tables.

The call to these routines accepts a SELECT statement that specifies the rows that you want to extract. Use the unformatted event table column definitions to assist with composing your SELECT statement.

There is no automatic purging of the event data written to an unformatted event table. You must manually purge data from the table. The column definitions for the

unformatted event table are useful when you want to purge a targeted set of records. Another option is to remove all the table rows using the TRUNCATE TABLE statement.

As part of the CREATE EVENT MONITOR statement, you can specify what to name the associated unformatted event table. If not specified, the name defaults to the same name as the event monitor. The SYSCAT.EVENTTABLES catalog view lists event monitors, their associated unformatted table, and other details.

The table below describes the columns in the unformatted event table. The key column is the event_data column. The other columns represent identifiers that you can use to locate events of interest. For further attributes of table columns, issue a DESCRIBE statement.

Table 3. Unformatted event table column definitions

Column name	Column data type	Column description
appl_id	VARCHAR	The identifier of the application within which the event occurred. A NULL value indicates that the application ID was not available.
appl_name	VARCHAR	The name of the application within which the event occurred. A NULL value indicates that the application name was not available.
event_correlation_id	BIT DATA	An optional event correlation ID. A NULL value indicates that the event correlation ID was not available. The value is based on the event monitor type: • LOCKING - Reserved for future use • UOW- Reserved for future use
event_data	BLOB	The entire event record data for an event captured by the event monitor, stored in its original binary form.

Table 3. Unformatted event table column definitions (continued)

Column name	Column data type	Column description
event_id	INTEGER	For locking event monitor records, an event identifer that is unique across the database. The ID is recycled at database activation time. Uniqueness is guaranteed by the combination of **event_timestamp**, **event_id**, **member**, and **event_type**. For UOW event monitor records, an alias of the UOW ID that is unique per connection. Uniqueness is guaranteed by the combination of **event_timestamp**, **event_id**, **member**, **event_type** and **appl_id**.
event_timestamp	TIMESTAMP	The timestamp when the event was generated by the event monitorAll child records will share the same timestamp as the parent record.
event_type	VARCHAR	The event type that occurred at the member of detection.
member	SMALLINT	The member where the event occurred.
partitioning_key	INTEGER	The partitioning key for the table, so that insert operations are performed locally on the database partition where the event monitor is running.
record_seq_num	INTEGER	The sequence number of the record that is stored within the event_data column.
record_type	INTEGER	The type of record that is stored within the event_data column.
service_subclass_name	VARCHAR	The name of the service subclass within which the event occurred. A NULL value indicates that the service subclass name was not available.
service_superclass_name	VARCHAR	The name of the service superclass within which the event occurred. A NULL value indicates that the service superclass name was not available.

Table 3. Unformatted event table column definitions (continued)

Column name	Column data type	Column description
workload_name	VARCHAR	The name of the workload within which the event occurred. A NULL value indicates that the workload name was not available.

db2evmonfmt tool for reading event monitor data

The Java-based, generic XML parser tool, **db2evmonfmt**, produces a readable flat-text output (text version) or a formatted XML output from the data generated by an event monitor that uses the unformatted event table. Based on the parameters that you specify, the **db2evmonfmt** tool determines how to parse the event monitor data and the type of output to create.

The **db2evmonfmt** tool is provided as Java source code. You must setup and compile this tool, before you can use it, by performing the following steps:

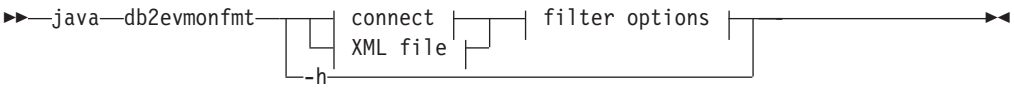
1. Locate the source code in the `sqllib/samples/java/jdbc` directory
2. Follow the instructions embedded in the Java source file to setup and compile the tool

You can modify the source code to change the output to your liking.

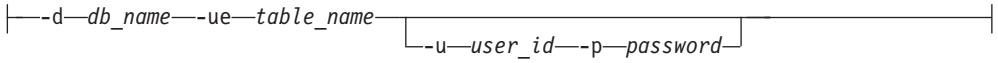
The tool uses XSLT style sheets to transform the event data into formatted text. You do not need to understand these style sheets. The tool will automatically load the correct style sheet, based on the event monitor type, and transform the event data. Each event monitor will provide default style sheets within the `sqllib/samples/xml/data` directory. The tool will also provide the following filtering options:

- Event ID
- Event timestamp
- Event type
- Workload name
- Service class name
- Application name

Tool syntax

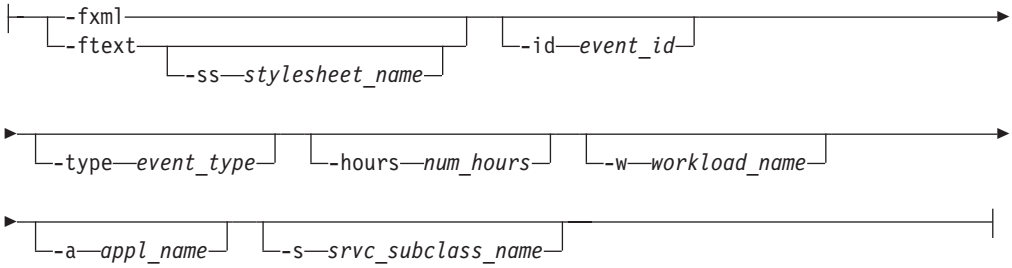


connect:



XML file:

filter options:



Tool parameters

java
To run the **db2evmonfmt** Java-based tool successfully, the **java** keyword must precede the tool name. The proper Java version to successfully run this tool is installed from the sqllib/java/jdk64 directory during the DB2 product installation.

-d *db_name*
Specifies the database name to which a connection is made

-ue *table_name*
Specifies the name of the unformatted event table

-u *user_id*
Specifies the user ID

-p *password*
Specifies the password

-f *xml_filename*
Specifies the name of the input XML file to format

-fxml
Produces a formatted XML document (pipe to stdout)

-ftext
Formats an XML document to a text document (pipe to stdout)

-ss *stylesheet_name*
Specifies the XSLT style sheet to use to transform the XML document

-id *event_id*
Displays all events matching the specified event ID

-type *event_type*
Displays all events matching the specified event type

-hours *num_hours*
Displays all events that have occurred within the specified last number of hours

-w *workload_name*
Displays all events that are part of the specified workload

-a *appl_name*
Displays all events that are part of the specified application

-s *srvc_subclass_name*
Displays all events that are part of the specified service subclass

XSLT style sheets

The DB2 database manager provides default XSLT style sheets (see Table 1) which can be found in the `sqllib/samples/java/jdbc` directory. You can change these style sheets to produce the desired output.

Table 4. Default XSLT style sheets for event monitors

Event monitor	Default XSLT style sheet
Locking	DB2EvmonLocking.xsl
Unit of work	DB2EvmonUOW.xsl

You can create your own XSLT style sheet to transform XML documents. You can pass these style sheets into the Java-based tool using the `-ss stylesheet_name` option.

Examples

Example 1

To obtain a formatted text output for all events that have occurred in the last 32 hours from the package cache unformatted event table PKG in database SAMPLE, issue the following command:

```
java db2evmonfmt -d sample -ue pkg -ftext -hours 32
```

Example 2

To obtain a formatted text output for all events of type LOCKTIMEOUT that have occurred in the last 24 hours from unformatted event table LOCK in database SAMPLE, issue the following command:

```
java db2evmonfmt -d sample -ue LOCK -ftext -hours 24 –type locktimeout
```

Example 3

To obtain a formatted text output from the XML source file LOCK.XML, extracting all events that match the event type LOCKWAIT in the last 5 hours, issue the following command:

```
java db2evmonfmt -f lock.xml -ftext -type lockwait -hours 5
```

Example 4

To obtain a formatted text output using the created XSLT style sheet SUMMARY.XSL for all events in the unformatted event table UOW in database SAMPLE, issue the following command:

```
java db2evmonfmt -d sample -ue uow -ftext -ss summary.xsl
```

Sample formatted flat-text output

The following sample of formatted flat-text output was generated from the locking event monitor XSLT style sheet:

```
------------------------------------------------
Event Entry     : 0
Event ID        : 1
Event Type      : Locktimeout
Event Timestamp : 2008-05-23-12.00.14.132329000
------------------------------------------------

Lock Details
------------
Lock Name       : 02000401000000000000000054
Lock Type       : Table
Lock Attributes : 00000000
Lock Count      : 1
```

```
Lock Hold Count : 0
Lock rrIID      : 0
Lock Status     : Waiting
Cursor Bitmap   : 00000000
Tablespace Name : USERSPACE1
Table Name      : NEWTON  .SARAH

Attributes           Requestor                      Holder
----------           ---------                      ------
Application Handle   [0-35]                         [0-16]
Application ID       *LOCAL.horton.080523160016    *LOCAL.horton.080523155938
Application Name     xaplus0001                     db2bp
Authentication ID    NEWTON                         HORTON
Requesting Agent     65                             21
Coordinating Agent   65                             21
Application Status   SQLM_CONNECTPEND               SQLM_CONNECTPEND
Lock Timeout         5000                           0
Workload Name        XAPLUS0010_WL02                SYSDEFAULTUSERWORKLOAD
Service Subclass     XAPLUS0010_SC02                SYSDEFAULTSUBCLASS
Current Request      Execute                        Execute Immediate
Lock Mode            Intent Exclusive               Exclusive
tpmon Userid
tpmon Wkstn
tpmon App
tpmon Accstring

Lock Requestor Current Activities
---------------------------------
Activity ID     : 2
Uow ID          : 1
Package ID      : 65426E4D4B584659
Package SectNo  : 3
Package Name    : NEWTON
Package Schema  : AKINTERF
Package Version :
Reopt           : always
Eff Isolation   : Cursor Stability
Eff Locktimeout : 5
Eff Degree      : 0
Nesting Level   : 0
Stmt Unicode    : No
Stmt Flag       : Dynamic
Stmt Type       : DML, Insert/Update/Delete
Stmt Text       : INSERT INTO SARAH VALUES(:H00008, :H00013, :H00014)


Lock Requestor Past Activities
------------------------------
Activity ID     : 1
Uow ID          : 1
Package ID      : 65426E4D4B584659
Package SectNo  : 2
Package Name    : NEWTON
Package Schema  : AKINTERF
Package Version :
Reopt           : always
Eff Isolation   : Cursor Stability
Eff Locktimeout : 5
Eff Degree      : 0
Nesting Level   : 0
Stmt Unicode    : No
Stmt Flag       : Dynamic
Stmt Type       : DML, Insert/Update/Delete
Stmt Text       : INSERT INTO NADIA VALUES(:H00007)


Lock Holder Current Activities
```

```
-----------------------------

Lock Holder Past Activities
-----------------------------
Activity ID     : 1
Uow ID          : 2
Package ID      : 41414141414E4758
Package SectNo  : 201
Package Name    : NULLID
Package Schema  : SQLC2G13
Package Version :
Reopt           : none
Eff Isolation   : Cursor Stability
Eff Locktimeout : 5
Eff Degree      : 0
Nesting Level   : 0
Stmt Unicode    : No
Stmt Flag       : Dynamic
Stmt Type       : DML, Select (blockable)
Stmt Text       : select * from newton.sarah

Activity ID     : 2
Uow ID          : 2
Package ID      : 41414141414E4758
Package SectNo  : 203
Package Name    : NULLID
Package Schema  : SQLC2G13
Package Version :
Reopt           : none
Eff Isolation   : Cursor Stability
Eff Locktimeout : 5
Eff Degree      : 0
Nesting Level   : 0
Stmt Unicode    : No
Stmt Flag       : Dynamic
Stmt Type       : DML, Lock Table
Stmt Text       : lock table newton.sarah in exclusive mode


-------------------------------------------------
Event Entry     : 1
Event ID        : 2
Event Type      : Locktimeout
Event Timestamp : 2008-05-23-12.04.42.144896000
-------------------------------------------------

...
...
...
```

Usage notes

The **db2evmonfmt** utility is a Java-based tool which must be preceded by the **java** keyword in order to run successfully. The Java version required is that which is installed with the DB2 product from the `sqllib/java/jdk64` directory.

Note: You can also use the EVMON_FORMAT_UE_TO_XML table function to format the binary events, contained in the unformatted event table BLOB column, into an XML document.

Monitoring database locking

Diagnosing and correcting lock contention situations in large DB2 environments can be complex and time consuming. The lock event monitor and other facilities are designed to simplify this task by collecting locking data.

Introduction

The lock event monitor is used to capture descriptive information about lock events at the time that they occur. The information captured identifies the key applications involved in the lock contention that resulted in the lock event. Information is captured for both the lock requestor (the application that received the deadlock or lock timeout error, or waited for a lock for more than the specified amount of time) and the current lock owner.

The information collected by the lock event monitor is written in binary format to an unformatted event table in the database. The captured data is processed in a post-capture step improving the efficiency of the capture process.

You can also directly access DB2 relational monitoring interfaces (table functions) to collect lock event information by using either dynamic or static SQL.

Determining if a deadlock or lock timeout has occurred is also simplified. Messages are written to the administration notification log when either of these events occurs; this supplements the SQL0911N (sqlcode -911) error returned to the application. In addition, a notification of lock escalations is also written to the administration notification log; this information can be useful in adjusting the size of the lock table and the amount of the table an application can use. There are also counters for lock timeouts (**lock_timeouts**), lock waits (**lock_waits**), and deadlocks (**deadlocks**) that can be checked.

The types of activities for which locking data can be captured include the following:

- SQL statements, such as:
  - DML
  - DDL
  - CALL
- **LOAD** command
- **REORG** command
- **BACKUP DATABASE** command
- Utility requests

The lock event monitor replaces the deprecated deadlock event monitors (CREATE EVENT MONITOR FOR DEADLOCKS statement and DB2DETAILDEADLOCK) and the deprecated lock timeout reporting feature (DB2_CAPTURE_LOCKTIMEOUT registry variable) with a simplified and consistent interface for gathering locking event data, and adds the ability to capture data on lock waits.

Functional overview

Two steps are required to enable the capturing of lock event data using the locking event monitor:

1. You must create a LOCK EVENT monitor using the CREATE EVENT MONITOR FOR LOCKING statement. You provide a name for the monitor and the name of an unformatted event table into which the lock event data will be written.
2. You must specify the level for which you want lock event data captured by using one of the following methods:

- You can specify particular workloads by either altering an existing workload, or by creating a new workload using the CREATE or ALTER WORKLOAD statements. At the workload level you must specify the type of lock event data you want captured (deadlock, lock timeout or lock wait), and whether you want the SQL statement history and input values for the applications involved in the locking. For lock waits you must also specify the amount of time that an application will wait for a lock, after which data is captured for the lock wait.
- You can collect data at the database level and affect all DB2 workloads by setting the appropriate database configuration parameter:

  mon_lockwait
  This parameter controls the generation of lock wait events

  Best practice is to enable lock wait data collection at the workload level.

  mon_locktimeout
  This parameter controls the generation of lock timeout events

  Best practice is to enable lock timeout data collection at the database level if they are unexpected by the application. Otherwise enable at workload level.

  mon_deadlock
  This parameter controls the generation of deadlock events

  Best practice is to enable deadlock data collection at the database level.

  mon_lw_thresh
  This parameter controls the amount of time spent in lock wait before an event for **mon_lockwait** is generated

The capturing of SQL statement history and input values incurs additional overhead, but this level of detail is often needed to successfully debug a locking problem.

After a locking event has occurred, the binary data in the unformatted event table can be transformed into an XML or a text document using a supplied Java-based application called **db2evmonfmt**. In addition, you can format the binary event data in the unformatted event table BLOB column into either an XML report document, using the EVMON_FORMAT_UE_TO_XML table function, or into a relational table, using the EVMON_FORMAT_UE_TO_TABLES procedure.

To aid in the determination of what workloads should be monitored for locking events, the administration notification log can be reviewed. Each time a deadlock or lock timeout is encountered, a message is written to the log. These messages identify the workload in which the lock requestor and lock owner or owners are running, and the type of locking event. There are also counters at the workload level for lock timeouts (**lock_timeouts**), lock waits (**lock_waits**), and deadlocks (**deadlocks**) that can be checked.

Information collected for a locking event

Some of the information for lock events collected by the lock event monitor include the following:

- The lock that resulted in an event
- The application holding the lock that resulted in the lock event

- The applications that were waiting for or requesting the lock that result in the lock event
- What the applications were doing during the lock event

Limitations

- There is no automatic purging of the lock event data written to the unformatted event table. You must periodically purge data from the table.
- You can output the collected event monitor data to only the unformatted event table. Outputs to file, pipe, and table are not supported.
- It is suggested that you create only one locking event monitor per database. Each additional event monitor only creates a copy of the same data.

Deprecated lock monitoring functionality

The deprecated detailed deadlock event monitor, DB2DETAILDEADLOCK, is created by default for each database and starts when the database is activated. The DB2DETAILDEADLOCK event monitor must be disabled and removed, otherwise both the deprecated and new event monitors will be collecting data and will significantly affect performance.

To remove the DB2DETAILDEADLOCK event monitor, issue the following SQL statements:

```
SET EVENT MONITOR DB2DETAILDEADLOCK state 0
DROP EVENT MONITOR DB2DETAILDEADLOCK
```

Collecting lock event data and generating reports

You can use the lock event monitor to collect lock timeout, lock wait, and deadlock information to help identify and resolve locking problems. After the lock event data has been collected in an unreadable form in an unformatted event table, this task describes how to obtain a readable text report.

Before you begin

To create the locking event monitor and collect lock event monitor data, you must have DBADM, or SQLADM authority.

About this task

The lock event monitor collects relevant information that helps with the identification and resolution of locking problems. For example, some of the information the lock event monitor collects for a lock event is as follows:

- The lock that resulted in a lock event
- The applications requesting or holding the lock that resulted in a lock event
- What the applications were doing during the lock event

This task provides instructions for collecting lock event data for a given workload. You might want to collect lock event data under the following conditions:

- You notice that lock wait values are longer than usual when using the MON_GET_WORKLOAD table function.
- An application returns a -911 SQL return code with reason code 68 in the administration notification log, stating that "The transaction was rolled back due to a lock timeout." See also message SQL0911N for further details.
- You notice a deadlock event message in the administration notification log (-911 SQL return code with reason code 2, stating that "The transaction was rolled

back due to a deadlock."). The log message indicates that the lock event occurred between two applications, for example, Application A and B, where A is part of workload FINANCE and B is part of workload PAYROLL. See also message SQL0911N for further details.

Restrictions

To view data values, you need the EXECUTE privilege on the EVMON_FORMAT_UE_* routines, which the SQLADM and DBADM authorities hold implicitly. You also need SELECT privilege on the unformatted event table table, which by default is held by users with the DATAACCESS authority and by the creator of the event monitor and the associated unformatted event table.

Procedure

To collect detailed information regarding potential future lock events, perform the following steps:

1. Create a lock event monitor called `lockevmon` by using the CREATE EVENT MONITOR FOR LOCKING statement, as shown in the following example:

   ```
   CREATE EVENT MONITOR lockevmon FOR LOCKING
      WRITE TO UNFORMATTED EVENT TABLE
   ```

   Note: The following lists important points to remember when creating an event monitor:

   - You can create event monitors ahead of time and not worry about using up disk space since nothing is written until you activate the data collection at the database or workload level
   - In a partitioned database environment, ensure that the event monitors are placed in a partitioned table space across all nodes. Otherwise, lock events will be missed at partitions where the partitioned table space is not present.
   - Ensure that you set up a table space and bufferpool to minimize the interference on high performance work caused by ongoing work during accesses to the tables to obtain data.

2. Activate the lock event monitor called `lockevmon` by running the following statement:

   ```
   SET EVENT MONITOR lockevmon STATE 1
   ```

3. To enable the lock event data collection at the workload level, issue the ALTER WORKLOAD statement with one of the following COLLECT clauses: COLLECT LOCK TIMEOUT DATA, COLLECT DEADLOCK DATA, or COLLECT LOCK WAIT DATA. Specify the WITH HISTORY option on the COLLECT clause. Setting the database configuration parameter affects the lock event data collection at the database level and all workloads are affected.

   For lock wait events

   To collect lock wait data for any lock acquired after 5 seconds for the FINANCE application and to collect lock wait data for any lock acquired after 10 seconds for the PAYROLL application, issue the following statements:

   ```
   ALTER WORKLOAD finance COLLECT LOCK WAIT DATA WITH HISTORY AND VALUES
      FOR LOCKS WAITING MORE THAN 5 SECONDS
   ALTER WORKLOAD payroll COLLECT LOCK WAIT DATA
      FOR LOCKS WAITING MORE THAN 10 SECONDS WITH HISTORY
   ```

   To set the **mon_lockwait** database configuration parameter with HIST_AND_VALUES input data value for the SAMPLE database, and

to set the **mon_lw_thresh** database configuration parameter for 10 seconds, issue the following commands:

```
db2 update db cfg for sample using mon_lockwait hist_and_values
db2 update db cfg for sample using mon_lw_thresh 10000000
```

For lock timeout events

To collect lock timeout data for the FINANCE and PAYROLL applications, issue the following statements:

```
ALTER WORKLOAD finance COLLECT LOCK TIMEOUT DATA WITH HISTORY
ALTER WORKLOAD payroll COLLECT LOCK TIMEOUT DATA WITH HISTORY
```

To set the **mon_locktimeout** database configuration parameter with HIST_AND_VALUES input data value for the SAMPLE database, issue the following command:

```
db2 update db cfg for sample using mon_locktimeout hist_and_values
```

For deadlock events

To collect data for the FINANCE and PAYROLL applications, issue the following statements:

```
ALTER WORKLOAD finance COLLECT DEADLOCK DATA WITH HISTORY
ALTER WORKLOAD payroll COLLECT DEADLOCK DATA WITH HISTORY
```

To set the **mon_deadlock** database configuration parameter with HIST_AND_VALUES input data value for the SAMPLE database, issue the following command:

```
db2 update db cfg for sample using mon_deadlock hist_and_values
```

4. Rerun the workload in order to receive another lock event notification.
5. Connect to the database.
6. Obtain the locking event report using one of the following approaches:
   a. Use the XML parser tool, **db2evmonfmt**, to produce a flat-text report based on the event data collected in the unformatted event table and using the default stylesheet, for example:

      ```
      java db2evmonfmt -d db_name -ue table_name -ftext -u user_id -p password
      ```

   b. Use the EVMON_FORMAT_UE_TO_XML table function to obtain an XML document.
   c. Use the EVMON_FORMAT_UE_TO_TABLES procedure to output the data into a relational table.
7. Analyze the report to determine the reason for the lock event problem and resolve it.
8. Turn OFF lock data collection for both FINANCE and PAYROLL applications by running the following statements or resetting the database configuration parameters:

   For lock wait events

   ```
   ALTER WORKLOAD finance COLLECT LOCK WAIT DATA NONE
   ALTER WORKLOAD payroll COLLECT LOCK WAIT DATA NONE
   ```

   To reset the **mon_lockwait** database configuration parameter with the default NONE input data value for the SAMPLE database, and to reset the **mon_lw_thresh** database configuration parameter back to its default value of 5 seconds, issue the following command:

   ```
   db2 update db cfg for sample using mon_lockwait none
   db2 update db cfg for sample using mon_lw_thresh 5000000
   ```

For lock timeout events

```
ALTER WORKLOAD finance COLLECT LOCK TIMEOUT DATA NONE
ALTER WORKLOAD payroll COLLECT LOCK TIMEOUT DATA NONE
```

To reset the **mon_locktimeout** database configuration parameter with the default NONE input data value for the SAMPLE database, issue the following command:

```
db2 update db cfg for sample using mon_locktimeout none
```

For deadlock events

```
ALTER WORKLOAD finance COLLECT DEADLOCK DATA NONE
ALTER WORKLOAD payroll COLLECT DEADLOCK DATA NONE
```

To reset the **mon_deadlock** database configuration parameter with the default WITHOUT_HIST input data value for the SAMPLE database, issue the following command:

```
db2 update db cfg for sample using mon_deadlock without_hist
```

What to do next

Rerun the application or applications to ensure that the locking problem has been eliminated.

Information written to XML for a locking event monitor

Information written for a locking event monitor from the EVMON_FORMAT_UE_TO_XML table function. This is also documented in the `sqllib/misc/DB2EvmonLocking.xsd` file.

db2_lock_event

The main schema that describes a lock timeout, lock wait or deadlock event in details.

Element content: ((db2_deadlock_graph {zero or one times (?)} , db2_participant {one or more (+)}) | (db2_message , db2_event_file))

Attributes:

QName	Type	Fixed	Default	Use	Annotation
id	xs:long			required	
type	xs:string - Max length: 32 (LOCKTIMEOUT, DEADLOCK, LOCKWAIT)			required	
timestamp	xs:dateTime			required	
member	member_type			required	
release	xs:long			required	
ANY attribute from ANY namespace					

db2_deadlock_graph

Schema element represents the DB2 Deadlock Graph. The graph outlines all the participants involved in the deadlock.

Contained by:db2_lock_event

Element content: (db2_participant {one or more (+)})

Attributes:

QName	Type	Fixed	Default	Use	Annotation
dl_conns	xs:int			required	
rolled_back_ participant_no	xs:int			required	
type	xs:string - Max length: 10 (local, global)			required	
ANY attribute from ANY namespace					

db2_participant

Schema element represents the application information of the all the participants involved in a lock event.

Contained by:db2_lock_event db2_deadlock_graph

Element content: (db2_object_requested {zero or one times (?)} , db2_app_details , db2_activity {zero or more (*)})

Attributes:

QName	Type	Fixed	Default	Use	Annotation
no	xs:int			required	
type	xs:string - Max length: 32 (Requester, Owner)			required	
participant_no_ holding_lk	xs:int			optional	
ANY attribute from ANY namespace					

db2_message

Error message

Contained by: db2_lock_event

Element content:

Type	Facet
xs:string	Max length: 1024

db2_event_file

Fully qualified path to file where event has been written.

Contained by: db2_lock_event

Element content:

Type	Facet
xs:string	Max length: 1024

application_handle

A system-wide unique ID for the application. See monitor element “agent_id - Application handle (agent ID) monitor element” on page 506 for more details.

Contained by: db2_app_details

Element content:

Type	Facet
xs:long	

appl_id

This identifier is generated when the application connects to the database at the database manager. See monitor element “appl_id - Application ID monitor element” on page 519 for more details.

Contained by:db2_app_details

Element content:

Type	Facet
xs:string	Max length: 128

appl_name

The name of the application running at the client, as known to the database. See monitor element “appl_name - Application name monitor element” on page 523 for more details.

Contained by: db2_app_details

Element content:

Type	Facet
xs:string	Max length: 128

auth_id

The authorization ID of the user who invoked the application that is being monitored. See monitor element “auth_id - Authorization ID” on page 537 for more details.

Contained by: db2_app_details

Element content:

Type	Facet
xs:string	Max length: 128

agent_tid

The unique identifier for the engine dispatchable unit (EDU) for the agent. See monitor element “agent_pid - Engine dispatchable unit (EDU) identifier monitor element” on page 509 for more details.

Contained by:db2_app_details

Element content:

Type	Facet
xs:long	

coord_agent_tid

The engine dispatchable unit (EDU) identifier of the coordinator agent for the application. See monitor element “coord_agent_pid - Coordinator agent identifier monitor element” on page 589 for more details.

Contained by: db2_app_details

Element content:

Type	Facet
xs:long	

agent_status

The current status of the application. See monitor element “appl_status - Application Status” on page 525 for more details.

Contained by:db2_app_details

Element content:

Type	Facet
xs:string	Max length: 32

Attributes:

QName	Type	Fixed	Default	Use	Annotation
id	xs:int			optional	

appl_action

The action/request that the client application is performing.

Contained by:db2_app_details

Element content:

Type	Facet
xs:string	Max length: 32

Attributes:

QName	Type	Fixed	Default	Use	Annotation
id	xs:int			optional	

lock_timeout_val

The database configuration parameter lock timeout. Value in seconds. See monitor element "lock_timeout_val - Lock timeout value monitor element" on page 731 for more details.

Contained by:db2_app_details

Element content:

Type	Facet
xs:long	

lock_wait_val

The lock wait parameter in effect during the lock event. This is either the database configuration parameter **mon_lw_thresh** or the COLLECT LOCK WAIT DATA setting specified at the workload level. Value in milliseconds.

Contained by: db2_app_details

Element content:

Type	Facet
xs:long	

tentry_state

TEntry state. Internal use only.

Contained by: db2_app_details

Element content:

Type	Facet
xs:string	Max length: 32

Attributes:

QName	Type	Fixed	Default	Use	Annotation
id	xs:int			optional	

tentry_flag1

TEntry flags1. Internal use only.

Contained by: db2_app_details

Element content:

Type	Facet
xs:hexBinary	Max length: 8

tentry_flag2

TEntry flags2. Internal use only.

Contained by: db2_app_details

Element content:

Type	Facet
xs:hexBinary	Max length: 8

xid

XID - Global transaction identifier

Contained by: db2_app_details

Element content:

Type	Facet
xs:hexBinary	Max length: 140

workload_id

ID of the workload to which this application belongs. See monitor element "workload_id - Workload ID monitor element" on page 1063 for more details.

Contained by: db2_app_details

Element content:

Type	Facet
xs:int	

workload_name

Name of the workload to which this application belongs. See monitor element "workload_name - Workload name monitor element" on page 1064 for more details.

Contained by: db2_app_details

Element content:

Type	Facet
xs:string	Max length: 128

service_class_id

ID of the service subclass to which this application belongs. See monitor element "service_class_id - Service class ID monitor element" on page 907 for more details.

Contained by: db2_app_details

Element content:

Type	Facet
xs:int	

service_subclass_name

Name of the service subclass to which this application belongs. See monitor element "service_subclass_name - Service subclass name monitor element" on page 908 for more details.

Contained by: db2_app_details

Element content:

Type	Facet
xs:string	Max length: 128

current_request

The operation currently being processed or most recently processed.

Contained by:db2_app_details

Element content:

Type	Facet
xs:string	Max length: 32

lock_escalation

Indicates whether a lock request was made as part of a lock escalation. See monitor element “lock_escalation - Lock escalation monitor element” on page 720 for more details. Possible values: Yes or No.

Contained by: db2_app_details

Element content:

Type	Facet
xs:string	Max length: 3

past_activities_wrapped

Indicates whether the activities list has wrapped. The default limit on the number of past activities to be kept by any one application is 250. This default can be overridden using the registry variable **DB2_MAX_INACT_STMTS**. Users may want to choose a different value for the limit to increase or reduce the amount of system monitor heap used for inactive statement information.

Contained by: db2_app_details

Element content:

Type	Facet
xs:string	Max length: 3

client_userid

The client user ID generated by a transaction manager and provided to the server. See monitor element “client_userid - Client user ID monitor element” on page 562 for more details.

Contained by:db2_app_details

Element content:

Type	Facet
xs:string	Max length: 255

client_wrkstnname

Identifies the client system or workstation, if the sqleseti API was issued in this connection. See monitor element “client_wrkstnname - Client workstation name monitor element” on page 563 for more details.

Contained by: db2_app_details

Element content:

Type	Facet
xs:string	Max length: 255

client_applname

Identifies the server transaction program performing the transaction, if the sqleseti API was issued in this connection. See monitor element “client_applname - Client application name monitor element” on page 555 for more details.

Contained by: db2_app_details

Element content:

Type	Facet
xs:string	Max length: 255

client_acctng

The data passed to the target database for logging and diagnostic purposes, if the sqleseti API was issued in this connection. See monitor element “client_acctng - Client accounting string monitor element” on page 554 for more details.

Contained by: db2_app_details

Element content:

Type	Facet
xs:string	Max length: 255

db2_object_requested

Schema element represents the DB2 lock that the Requestor is attempting to acquire, which is being held by the Owner.

Contained by: db2_participant

Element content: ((lock_name , lock_object_type , lock_specifics , lock_attributes , lock_current_mode , lock_mode_requested , lock_mode , lock_count , lock_hold_count , lock_rriid , lock_status , lock_release_flags , tablespace_name , table_name , table_schema , lock_object_type_id , lock_wait_start_time , lock_wait_end_time , ANY content (skip) {zero or more (*)}) | (threshold_name , threshold_id , queued_agents , queue_start_time , ANY content (skip) {zero or more (*)}))

Attributes:

QName	Type	Fixed	Default	Use	Annotation
type	xs:string - Max length: 10 (lock, ticket)			required	

db2_app_details

Schema element represents the details regarding this participant.

Contained by: db2_participant

Element content: (application_handle , appl_id , appl_name , auth_id , agent_tid , coord_agent_tid , agent_status , appl_action , lock_timeout_val , lock_wait_val , tentry_state , tentry_flag1 , tentry_flag2 , xid , workload_id , workload_name , service_class_id , service_subclass_name , current_request , lock_escalation , past_activities_wrapped , client_userid , client_wrkstnname , client_applname , client_acctng , ANY content (skip) {zero or more (*)})

db2_activity

List of all DB2 activities the application is currently executing or has executed.

Contained by: db2_participant

Element content: (db2_activity_details , db2_input_variable {zero or more (*)})

Attributes:

QName	Type	Fixed	Default	Use	Annotation
type	xs:string - Max length: 10 (current, past)			required	
ANY attribute from ANY namespace					

lock_name

Internal binary lock name. This element serves as a unique identifier for locks. See monitor element “lock_name - Lock name monitor element” on page 726 for more details.

Contained by: db2_object_requested

Element content:

Type	Facet
xs:hexBinary	Max length: 16

lock_object_type

The type of object the application is waiting to obtain a lock. See monitor element "lock_object_type - Lock object type waited on monitor element" on page 728 for more details.

Contained by: db2_object_requested

Element content:

Type	Facet
xs:string	Max length: 32

Attributes:

QName	Type	Fixed	Default	Use	Annotation
id	xs:long			optional	

lock_specifics

Internal specifics about the lock. For information use only.

Contained by: db2_object_requested

Element content:

Type	Facet
xs:string	Max length: 128

lock_attributes

Lock attributes. See monitor element "lock_attributes - Lock attributes monitor element" on page 717 for more details.

Contained by: db2_object_requested

Element content:

Type	Facet
xs:hexBinary	Max length: 4

lock_current_mode

Original lock before conversion. See monitor element "lock_current_mode - Original lock mode before conversion monitor element" on page 719 for more details.

Contained by:db2_object_requested

Element content:

Type	Facet
xs:string	Max length: 32

Attributes:

QName	Type	Fixed	Default	Use	Annotation
id	xs:long			optional	
mode	db2_string_type_6			optional	

lock_mode_requested

The lock mode being requested by this participant. See monitor element "lock_mode_requested - Lock mode requested monitor element" on page 725 for more details.

Contained by:db2_object_requested

Element content:

Type	Facet
xs:string	Max length: 32

Attributes:

QName	Type	Fixed	Default	Use	Annotation
id	xs:long			optional	
mode	db2_string_type_6			optional	

lock_mode

The type of lock being held. See monitor element "lock_mode - Lock mode monitor element" on page 724 for more details.

Contained by:db2_object_requested

Element content:

Type	Facet
xs:string	Max length: 32

Attributes:

QName	Type	Fixed	Default	Use	Annotation
id	xs:long			optional	
mode	db2_string_type_6			optional	

lock_count

The number of locks on the lock being held. See monitor element "lock_count - Lock count monitor element" on page 718 for more details.

Contained by:db2_object_requested

Element content:

Type	Facet
xs:long	

lock_hold_count

The number of holds placed on the lock. See monitor element "lock_hold_count - Lock hold count monitor element" on page 723 for more details.

Contained by: db2_object_requested

Element content:

Type	Facet
xs:long	

lock_rriid

IID for Row locking. Internal use only.

Contained by: db2_object_requested

Element content:

Type	Facet
xs:long	

lock_status

Indicates the internal status of the lock. See monitor element "lock_status - Lock status monitor element" on page 730 for more details.

Contained by: db2_object_requested

Element content:

Type	Facet
xs:string	Max length: 32

Attributes:

QName	Type	Fixed	Default	Use	Annotation
id	xs:int			optional	

lock_release_flags

Lock release flags. See monitor element "lock_release_flags - Lock release flags monitor element" on page 730 for more details.

Contained by: db2_object_requested

Element content:

Type	Facet
xs:hexBinary	Max length: 4

tablespace_name

The name of the table space where the lock is held. See monitor element "tablespace_name - Table space name monitor element" on page 961 for more details.

Contained by: db2_object_requested

Element content:

Type	Facet
xs:string	Max length: 128

Attributes:

QName	Type	Fixed	Default	Use	Annotation
id	xs:long			optional	

table_name

The name of the table where the lock is held. See monitor element "table_name - Table name monitor element" on page 952 for more details.

Contained by: db2_object_requested

Element content:

Type	Facet
xs:string	Max length: 128

Attributes:

QName	Type	Fixed	Default	Use	Annotation
id	xs:long			optional	
data_member_id	member_type			optional	The identifier of the data member for which information is returned.

table_schema

The schema of the table. See monitor element "table_schema - Table schema name monitor element" on page 953 for more details.

Contained by: db2_object_requested

Element content:

Type	Facet
xs:string	Max length: 128

lock_object_type_id

The type of object the application is waiting to obtain a lock. See monitor element "lock_object_type - Lock object type waited on monitor element" on page 728 for more details.

Contained by: db2_object_requested

Element content:

Type	Facet
xs:hexBinary	Max length: 1

lock_wait_start_time

The data and time the application started waiting to obtain a lock on the object that is currently locked by the lock owner. See monitor element "lock_wait_start_time - Lock wait start timestamp monitor element" on page 734 for more details.

Contained by: db2_object_requested

Element content:

Type	Facet
xs:dateTime	

lock_wait_end_time

The data and time the application stopped waiting to obtain a lock on the object that is currently locked by the lock owner.

Contained by: db2_object_requested

Element content:

Type	Facet
xs:dateTime	

threshold_name

The name of the threshold queue.

Contained by: db2_object_requested

Element content:

Type	Facet
xs:string	Max length: 128

threshold_id

The ID of the threshold queue.

Contained by: db2_object_requested

Element content:

Type	Facet
xs:int	

queued_agents

The total number of agents currently queued in the threshold.

Contained by: db2_object_requested

Element content:

Type	Facet
xs:long	

queue_start_time

The data and time the application started waiting in the queue to obtain a threshold ticket.

Contained by: db2_object_requested

Element content:

Type	Facet
xs:dateTime	

db2_participant

Schema element represents a single stack entry in a deadlock graph.

Contained by: db2_lock_event db2_deadlock_graph

Attributes:

QName	Type	Fixed	Default	Use	Annotation
no	xs:int			required	
deadlock_member	member_type			required	
participant_no_ holding_lk	xs:int			required	
application_handle	application_handle_type			required	
ANY attribute from ANY namespace					

activity_id

Counter which uniquely identifies an activity for an application within a given unit of work. See monitor element "activity_id - Activity ID monitor element" on page 502 for more details.

Contained by: db2_activity_details

Element content:

Type	Facet
xs:int	

uow_id

The unit of work ID to which this activity record applies. See monitor element "uow_id - Unit of work ID monitor element" on page 1048 for more details.

Contained by: db2_activity_details

Element content:

Type	Facet
xs:int	

package_name

The name of the package that contains the SQL statement currently executing. See monitor element "package_name - Package name monitor element" on page 791 for more details.

Contained by: db2_activity_details

Element content:

Type	Facet
xs:string	Max length: 128

package_schema

The schema name of the package associated with an SQL statement. See monitor element "package_schema - Package schema monitor element" on page 792 for more details.

Contained by: db2_activity_details

Element content:

Type	Facet
xs:string	Max length: 128

package_version_id

The package version identifies the version identifier of the package that contains the SQL statement currently executing. See monitor element "package_version_id - Package version monitor element" on page 792 for more details.

Contained by: db2_activity_details

Element content:

Type	Facet
xs:string	Max length: 64

consistency_token

The package consistency token helps to identify the version of the package that contains the SQL statement currently executing. See monitor element "consistency_token - Package consistency token monitor element" on page 580 for more details.

Contained by: db2_activity_details

Element content:

Type	Facet
xs:string	Max length: 8

section_number

The internal section number in the package for the SQL statement currently processing or most recently processed. See monitor element "section_number - Section number monitor element" on page 901 for more details.

Contained by: db2_activity_details

Element content:

Type	Facet
xs:long	

reopt

The **REOPT** bind option used to precompile this package. Possible values are: NONE, ONCE, and ALWAYS. See the **REOPT** bind options for more details.

Contained by: db2_activity_details

Element content:

Type	Facet
xs:string	Max length: 10

incremental_bind

The package was incrementally bound at execution time. Possible values: Yes or No.

Contained by: db2_activity_details

Element content:

Type	Facet
xs:string	Max length: 3

effective_isolation

The isolation value in effect for the SQL statement while it was being run. See monitor element "effective_isolation - Effective isolation monitor element" on page 633 for more details.

Contained by: db2_activity_details

Element content:

Type	Facet
xs:string	Max length: 2

Attributes:

QName	Type	Fixed	Default	Use	Annotation
id	xs:long			optional	

effective_query_degree

The degree value in effect for the SQL statement while it was being run. See monitor element "effective_query_degree - Effective query degree monitor element" on page 634 for more details.

Contained by: db2_activity_details

Element content:

Type	Facet
xs:long	

stmt_unicode

The SQL statement Unicode flag. Possible values: Yes or No.

Contained by:db2_activity_details

Element content:

Type	Facet
xs:string	Max length: 3

stmt_lock_timeout

The locktimeout value in effect for the SQL statement while it was being run. See monitor element “stmt_lock_timeout - Statement lock timeout monitor element” on page 934 for more details.

Contained by: db2_activity_details

Element content:

Type	Facet
xs:int	

stmt_type

The type of SQL statement processed. Possible values: Dynamic or Static. See monitor element “stmt_type - Statement type monitor element” on page 942 for more details.

Contained by: db2_activity_details

Element content:

Type	Facet
xs:string	Max length: 10

Attributes:

QName	Type	Fixed	Default	Use	Annotation
id	xs:long			required	

stmt_operation

See monitor element “stmt_operation/operation - Statement operation monitor element” on page 935 for more details.

Contained by: db2_activity_details

Element content:

Type	Facet
xs:string	Max length: 128

stmt_query_id

Internal query identifier given to any SQL statement. See monitor element “stmt_query_id - Statement query identifier monitor element” on page 938 for more details.

Contained by: db2_activity_details

Element content:

Type	Facet
xs:long	

stmt_nest_level

This element shows the level of nesting or recursion in effect when the statement was run. See monitor element “stmt_nest_level - Statement nesting level monitor element” on page 935 for more details.

Contained by: db2_activity_details

Element content:

Type	Facet
xs:long	

stmt_invocation_id

This element shows the identifier of the routine invocation in which the SQL statement was run. See monitor element “stmt_invocation_id - Statement invocation identifier monitor element” on page 932 for more details.

Contained by: db2_activity_details

Element content:

Type	Facet
xs:long	

stmt_source_id

This element shows the internal identifier given to the source of the SQL statement that was run. See monitor element “stmt_source_id - Statement source identifier” on page 939 for more details.

Contained by: db2_activity_details

Element content:

Type	Facet
xs:long	

stmt_pkgcache_id

This element shows the internal package cache identifier of a dynamic SQL statement. See monitor element "stmt_pkgcache_id - Statement package cache identifier monitor element" on page 937 for more details.

Contained by: db2_activity_details

Element content:

Type	Facet
xs:long	

stmt_text

The text of the SQL statement. See monitor element "stmt_text - SQL statement text monitor element" on page 940 for more details.

Contained by: db2_activity_details

Element content:

Type	Facet
xs:string	Max length: 2097152

stmt_first_use_time

This element shows the first time the statement entry was processed. For cursor operations, "stmt_first_use_time - Statement first use timestamp monitor element" on page 931 shows when the cursor was opened. At application coordination nodes, this value reflects the application requests; at non-coordinator nodes, this value reflects when requests were received from the originating node. See monitor element **stmt_first_use_time** for more details.

Contained by: db2_activity_details

Element content:

Type	Facet
xs:dateTime	

stmt_last_use_time

This element shows the last time the statement entry was processed. For cursor operations, "stmt_last_use_time - Statement last use timestamp monitor element" on page 934

on page 934 shows the time of the last action on the cursor where that action could be an open, fetch, or close. At application coordination nodes, this value reflects the application requests; at non-coordinator nodes, this value reflects when requests were received from the originating node. See monitor element **stmt_last_use_time** for more details.

Contained by: db2_activity_details

Element content:

Type	Facet
xs:dateTime	

db2_activity_details

Schema represents the details regarding this activity.

Contained by: db2_activity

Element content: (activity_id , uow_id , package_name , package_schema , package_version_id , consistency_token , section_number , reopt , incremental_bind , effective_isolation , effective_query_degree , stmt_unicode , stmt_lock_timeout , stmt_type , stmt_operation , stmt_query_id , stmt_nest_level , stmt_invocation_id , stmt_source_id , stmt_pkgcache_id , stmt_text , stmt_first_use_time , stmt_last_use_time , ANY content (skip) {zero or more (*)})

db2_input_variable

Schema element represents the list of input variables associated with the SQL statement.

Contained by: db2_activity

Element content: (stmt_value_index , stmt_value_isreopt , stmt_value_isnull , stmt_value_type , stmt_value_data , ANY content (skip) {zero or more (*)})

stmt_value_index

The element represents the position of the input parameter marker or host variable used in the SQL statement. See monitor element “stmt_value_index - Value index” on page 944 for more details.

Contained by: db2_input_variable

Element content:

Type	Facet
xs:int	

stmt_value_isreopt

The element shows whether the variable was used during statement reoptimization. See monitor element “stmt_value_isreopt - Variable used for statement reoptimization monitor element” on page 945 for more details.

Contained by: db2_input_variable

Element content:

Type	Facet
xs:string	Max length: 3

Attributes:

QName	Type	Fixed	Default	Use	Annotation
id	xs:int			required	

stmt_value_isnull

The element shows whether a data value associated with the SQL statement is the NULL value. See monitor element “stmt_value_isnull - Value has null value monitor element” on page 945 for more details.

Contained by: db2_input_variable

Element content:

Type	Facet
xs:string	Max length: 3

Attributes:

QName	Type	Fixed	Default	Use	Annotation
id	xs:int			required	

stmt_value_type

The element contains a string representation of the type of data value associated with an SQL statement. See monitor element “stmt_value_type - Value type monitor element” on page 946 for more details.

Contained by:db2_input_variable

Element content:

Type	Facet
xs:string	Max length: 16

stmt_value_data

The element contains a string representation of a data value associated with an SQL statement. See monitor element “stmt_value_data - Value data” on page 944 for more details.

Contained by: db2_input_variable

Element content:

Type	Facet
xs:string	Max length: 32768

Information written to relational tables for a locking event monitor

Information written for a locking event monitor from the EVMON_FORMAT_UE_TO_TABLES table function. This is also documented in the `sqllib/misc/DB2EvmonLocking.xsd` file.

Table 5. Information returned for a locking event monitor: Table name: LOCK_EVENT

Column Name	Data Type	Description
XMLID	VARCHAR(1000) NOT NULL	
EVENT_ID	BIGINT NOT NULL	
EVENT_TYPE	VARCHAR(128) NOT NULL	
EVENT_TIMESTAMP	TIMESTAMP NOT NULL	
MEMBER	SMALLINT NOT NULL	"member - Database member monitor element" on page 760
DL_CONNS	INTEGER	"dl_conns - Connections involved in deadlock monitor element" on page 632
ROLLED_BACK_PARTICIPANT_NO	INTEGER	"rolled_back_participant_no - Rolled back application participant monitor element" on page 887

Table 6. Information returned for a locking event monitor: Table name: LOCK_PARTICIPANTS

Column Name	Data Type	Description
XMLID	VARCHAR(1000) NOT NULL	
PARTICIPANT_NO	INTEGER	"participant_no - Participant within Deadlock" on page 798
PARTICIPANT_TYPE	VARCHAR(10)	
PARTICIPANT_NO_HOLDING_LK	INTEGER	"participant_no_holding_lk - Participant Holding a Lock on the Object Required by Application" on page 799
APPLICATION_HANDLE	BIGINT	"application_handle - Application handle monitor element" on page 528
APPL_ID	VARCHAR(128)	"appl_id - Application ID monitor element" on page 519
APPL_NAME	VARCHAR(128)	"appl_name - Application name monitor element" on page 523
AUTH_ID	VARCHAR(128)	"auth_id - Authorization ID" on page 537
AGENT_TID	BIGINT	
COORD_AGENT_TID	BIGINT	

Table 6. Information returned for a locking event monitor: Table name: LOCK_PARTICIPANTS (continued)

Column Name	Data Type	Description
AGENT_STATUS	INTEGER	"appl_status - Application Status" on page 525
LOCK_TIMEOUT_VAL	BIGINT	"lock_timeout_val - Lock timeout value monitor element" on page 731
LOCK_WAIT_VAL	BIGINT	The amount of time spent in lock wait (in milliseconds) before an event for mon_lockwait is generated.
WORKLOAD_ID	INTEGER	"workload_id - Workload ID monitor element" on page 1063
WORKLOAD_NAME	VARCHAR(128)	"workload_name - Workload name monitor element" on page 1064
SERVICE_CLASS_ID	INTEGER	"service_class_id - Service class ID monitor element" on page 907
SERVICE_SUBCLASS_NAME	VARCHAR(128)	"service_subclass_name - Service subclass name monitor element" on page 908
CURRENT_REQUEST	VARCHAR(32)	
LOCK_ESCALATION	CHAR(3)	"lock_escalation - Lock escalation monitor element" on page 720
PAST_ACTIVITIES_WRAPPED	CHAR(3)	
CLIENT_USERID	VARCHAR(255)	"client_userid - Client user ID monitor element" on page 562
CLIENT_WRKSTNNAME	VARCHAR(255)	"client_wrkstnname - Client workstation name monitor element" on page 563
CLIENT_APPLNAME	VARCHAR(255)	"client_applname - Client application name monitor element" on page 555
CLIENT_ACCTNG	VARCHAR(255)	"client_acctng - Client accounting string monitor element" on page 554
OBJECT_REQUESTED	VARCHAR(10)	
LOCK_NAME	CHAR(32)	"lock_name - Lock name monitor element" on page 726
LOCK_OBJECT_TYPE	VARCHAR(32)	"lock_object_type - Lock object type waited on monitor element" on page 728
LOCK_OBJECT_TYPE_ID	CHAR(1) FOR BIT DATA	
LOCK_ATTRIBUTES	CHAR(8)	"lock_attributes - Lock attributes monitor element" on page 717
LOCK_CURRENT_MODE	BIGINT	"lock_current_mode - Original lock mode before conversion monitor element" on page 719

Table 6. Information returned for a locking event monitor: Table name: LOCK_PARTICIPANTS (continued)

Column Name	Data Type	Description
LOCK_MODE_REQUESTED	BIGINT	"lock_mode_requested - Lock mode requested monitor element" on page 725
LOCK_MODE	BIGINT	"lock_mode - Lock mode monitor element" on page 724
LOCK_COUNT	BIGINT	"lock_count - Lock count monitor element" on page 718
LOCK_HOLD_COUNT	BIGINT	"lock_hold_count - Lock hold count monitor element" on page 723
LOCK_RRIID	BIGINT	
LOCK_STATUS	BIGINT	"lock_status - Lock status monitor element" on page 730
LOCK_RELEASE_FLAGS	CHAR(8)	"lock_release_flags - Lock release flags monitor element" on page 730
LOCK_WAIT_START_TIME	TIMESTAMP	"lock_wait_start_time - Lock wait start timestamp monitor element" on page 734
LOCK_WAIT_END_TIME	TIMESTAMP	
TABLE_FILE_ID	BIGINT	"table_file_id - Table file ID monitor element" on page 951
TABLE_NAME	VARCHAR(128)	"table_name - Table name monitor element" on page 952
TABLE_SCHEMA	VARCHAR(128)	"table_schema - Table schema name monitor element" on page 953
TABLESPACE_NAME	VARCHAR(128)	"tablespace_name - Table space name monitor element" on page 961
THRESHOLD_ID	INTEGER	
THRESHOLD_NAME	VARCHAR(128)	"threshold_name - Threshold name monitor element" on page 984

Table 7. Information returned for a locking event monitor: Table name: LOCK_PARTICIPANT_ACTIVITIES

Column Name	Data Type	Description
XMLID	VARCHAR(1000) NOT NULL	
PARTICIPANT_NO	INTEGER	"participant_no - Participant within Deadlock" on page 798
ACTIVITY_ID	INTEGER	"activity_id - Activity ID monitor element" on page 502
ACTIVITY_TYPE	VARCHAR(10)	"activity_type - Activity type monitor element" on page 504
UOW_ID	INTEGER	"uow_id - Unit of work ID monitor element" on page 1048

Table 7. Information returned for a locking event monitor: Table name: LOCK_PARTICIPANT_ACTIVITIES (continued)

Column Name	Data Type	Description
PACKAGE_NAME	VARCHAR(128)	"package_name - Package name monitor element" on page 791
PACKAGE_SCHEMA	VARCHAR(128)	"package_schema - Package schema monitor element" on page 792
PACKAGE_VERSION_ID	VARCHAR(64)	"package_version_id - Package version monitor element" on page 792
CONSISTENCY_TOKEN	VARCHAR(8)	"consistency_token - Package consistency token monitor element" on page 580
SECTION_NUMBER	BIGINT	"section_number - Section number monitor element" on page 901
REOPT	VARCHAR(10)	
INCREMENTAL_BIND	CHAR(3)	
EFFECTIVE_ISOLATION	CHAR(2)	"effective_isolation - Effective isolation monitor element" on page 633
EFFECTIVE_QUERY_DEGREE	BIGINT	"effective_query_degree - Effective query degree monitor element" on page 634
STMT_LOCK_TIMEOUT	INTEGER	"stmt_lock_timeout - Statement lock timeout monitor element" on page 934
STMT_TYPE	BIGINT	"stmt_type - Statement type monitor element" on page 942
STMT_QUERY_ID	BIGINT	"stmt_query_id - Statement query identifier monitor element" on page 938
STMT_NEST_LEVEL	BIGINT	"stmt_nest_level - Statement nesting level monitor element" on page 935
STMT_INVOCATION_ID	BIGINT	"stmt_invocation_id - Statement invocation identifier monitor element" on page 932
STMT_SOURCE_ID	BIGINT	"stmt_source_id - Statement source identifier" on page 939
STMT_PKGCACHE_ID	BIGINT	"stmt_pkgcache_id - Statement package cache identifier monitor element" on page 937
STMT_FIRST_USE_TIME	TIMESTAMP	"stmt_first_use_time - Statement first use timestamp monitor element" on page 931
STMT_LAST_USE_TIME	TIMESTAMP	"stmt_last_use_time - Statement last use timestamp monitor element" on page 934

Table 7. Information returned for a locking event monitor: Table name: LOCK_PARTICIPANT_ACTIVITIES (continued)

Column Name	Data Type	Description
STMT_TEXT	CLOB(2097152)	"stmt_text - SQL statement text monitor element" on page 940

Table 8. Information returned for a locking event monitor: Table name: LOCK_ACTIVITY_VALUES

Column Name	Data Type	Description
XMLID	VARCHAR(1000) NOT NULL	
PARTICIPANT_NO	INTEGER	"participant_no - Participant within Deadlock" on page 798
ACTIVITY_ID	INTEGER	"activity_id - Activity ID monitor element" on page 502
UOW_ID	INTEGER	"uow_id - Unit of work ID monitor element" on page 1048
STMT_VALUE_INDEX	INTEGER	"stmt_value_index - Value index" on page 944
STMT_VALUE_ISREOPT	INTEGER	"stmt_value_isreopt - Variable used for statement reoptimization monitor element" on page 945
STMT_VALUE_ISNULL	INTEGER	"stmt_value_isnull - Value has null value monitor element" on page 945
STMT_VALUE_TYPE	CHAR(16)	"stmt_value_type - Value type monitor element" on page 946
STMT_VALUE_DATA	CLOB(32K)	"stmt_value_data - Value data" on page 944

Monitoring package cache events

The package cache event monitor captures data related to statement entries that have been flushed from the database package cache. This event monitor provides the history of the contents of the package cache which can help with SQL query performance and problem determination issues.

Overview

The package cache event monitor collects the same information as the MON_GET_PKG_CACHE_STMT table function, including the full set of available activity metrics and the executable section information of an entry.

Two control mechanisms, on the CREATE EVENT MONITOR statement, help limit the volume of data that can be captured. The two control mechanisms provide the following capabilities:

1. Filter entries with the WHERE clause based on one or more of the following:
   - Whether the last update of the metrics for an entry occurs after a specific time before it is evicted (UPDATED_SINCE_BOUNDARY_TIME). An entry will only be collected if the time that the metrics were last updated is more recent than boundary time defined for the event monitor. The boundary time for an event monitor can be set using the MON_GET_PKG_CACHE_STMT

table function. If no boundary time has been set for the event monitor, the UPDATED_SINCE_BOUNDARY_TIME clause will have no effect.
- The number of times the section of an entry was executed (NUM_EXECUTIONS)
- The total aggregated amount of time spent executing the statement (STMT_EXEC_TIME)

2. COLLECT DATA clause options:
   - COLLECT BASE DATA

     Same information collected as the MON_GET_PKG_CACHE_STMT table function, as well as the full set of available activity metrics
   - COLLECT DETAILED DATA

     Collects the same information gathered with the COLLECT BASE DATA clause and includes the executable section of the entry

When you need to investigate the individual execution of an SQL statement, you can use the MON_GET_PKG_CACHE_STMT table function (if the entries are still in the package cache) to compare the behavior of a cached entry relative to others. The execution metrics, compilation environment, and detailed descriptions for a cached entry are available for diagnostic purposes.

If an entry has already been flushed from the package cache, you can use the package cache event monitor to review the history of the cached entries which were flushed from the package cache. The history data contains the same information that the MON_GET_PKG_CACHE_STMT table function provides. In addition, the event monitor also provides the executable section of the statement. All of this applies to both dynamic and static SQL statements.

Creating a package cache event monitor

To create the package cache event monitor and collect package cache event monitor data, you must have DBADM or SQLADM authority.

The package cache event monitor only has the unformatted event table as the output option.

Before you create a package cache event monitor, identify the table space where you plan to store the unformatted event table for your event monitor. The recommended practice is to have a table space dedicated and configured to store the unformatted event table associated with any event monitor. Create the package cache event monitor in a tablespace with at least 8K pagesize to ensure that the event data is contained within the inlined BLOB column of the unformatted event table. If the BLOB column is not inlined, then the performance of writing and reading the events to the unformatted event table might not be efficient.

The database manager attempts to inline the event_data BLOB column in the unformatted event table, but this is not always possible. To check that the rows in the unformatted event table have been inlined, use the ADMIN_IS_INLINED function. If the rows have not been inlined, use the ADMIN_EST_INLINE_LENGTH functions to determine how much space the rows need.

Your other options, when you create an event monitor, are to specify any existing table space or to not specify any and have one chosen by default.

To setup a package cache event monitor using defaults and best practices, complete the following steps:

- Create the event monitor by issuing the CREATE EVENT MONITOR statement. The following example uses defaults where possible and specifies to store the unformatted event table in an existing table space MY_EVMON_TABLESPACE:

```
CREATE EVENT MONITOR MY_PKGCACHE_EVMON
  FOR PACKAGE CACHE
   WRITE TO UNFORMATTED EVENT TABLE (IN MY_EVMON_TABLESPACE)
```

Enabling data collection

To enable data collection, you must activate the event monitor using the SET EVENT MONITOR STATE statement. The package cache event monitor is not passive; following activation, it automatically starts collecting data whenever a statement is flushed from the package cache and meets the filter criteria set at the time of creation of the package cache event monitor.

Accessing event data captured by a package cache event monitor

This type of event monitor writes data in a binary format to an unformatted event table. You can access this data using the following table functions:

- EVMON_FORMAT_UE_TO_XML - extracts data from an unformatted event table into an XML document.
- EVMON_FORMAT_UE_TO_TABLES - extracts data from an unformatted event table into a set of relational tables.

The schema file `~/sqllib/misc/DB2EvmonPkgCache.xsd` is used to document the expected output of the package cache event monitor report in an XML document. The schema file will reference a common monitor schema file (`DB2MonCommon.xsd`) to avoid duplicating the common contents.

An XML stylesheet is provided in `~/sqllib/samples/jdbc/DB2EvmonPkgCache.xsl`.

Use these table functions to specify the data to extract using a SELECT statement. You have full control over selection, ordering, and other aspects provided by the SELECT statement.

You can also use the **db2evmonfmt** command to perform the following tasks:

- Select events of interest based on the following attributes: executable ID, section type, query cost estimate, statement package cache ID, and flush time.
- Choose whether to receive the output in the form of a text report or a formatted XML document.
- Control the output format by creating your own XSLT style sheets instead of using the ones provided by the **db2evmonfmt** command.

For example, the following command provides a package cache report that:

1. Selects package cache events that have occurred in the past 24 hours in the database SAMPLE. These event records are obtained from the unformatted event table called SAMPLE_PKGCACHE_EVENTS.
2. Provides formatted text output using the `DB2EvmonPkgCache.xsl` style sheet.

```
java db2evmonfmt -d SAMPLE -ue SAMPLE_PKGCACHE_EVENTS -ftext -ss DB2EvmonPkgCache.xsl -hours 24
```

Collecting package cache event data and generating reports

You can use the package cache event monitor to collect data about statement entries that were flushed from the database package cache. After the package cache event data has been collected in an unformatted event table, follow the directions in this task to obtain a text report.

Before you begin

To collect package cache event monitor data, you must have DBADM or SQLADM authority.

About this task

The package cache event monitor collects relevant history information about what was in the package cache to help with query performance and problem determination issues related to SQL statements. For example, some of the information the package cache event monitor collects from the database package cache is as follows:

- Executable ID (EXECUTABLE_ID)
- The estimated cost of the query (QUERY_COST_ESTIMATE)
- The time that the entry was flushed from the package cache (Event Timestamp)

This task provides instructions for collecting package cache event data.

Restrictions

Input data values are not viewable if you do not have DBADM or SQLADM authority.

Procedure

To collect detailed information regarding package cache events, perform the following steps:

1. Create a package cache event monitor called `cachestmtevmon` by using the CREATE EVENT MONITOR FOR PACKAGE CACHE statement, as shown in the following example:

   ```
   CREATE EVENT MONITOR cachestmtevmon FOR PACKAGE CACHE
      WRITE TO UNFORMATTED EVENT TABLE
   ```

2. Activate the package cache event monitor called `cachestmtevmon` by running the following statement:

   ```
   SET EVENT MONITOR cachestmtevmon STATE 1
   ```

3. Unlike the locking and the unit of work event monitors, the package cache event monitor automatically starts collecting data after the event monitor is activated.
4. Connect to the database.
5. Run the application, workload or SQL statements for which you want to collect event monitor information.
6. If you want to turn OFF package cache data collection, deactivate the event monitor by running the following command:

   ```
   SET EVENT MONITOR cachestmtevmon STATE 0
   ```

7. Obtain the package cache event report using the XML parser tool, **db2evmonfmt**, to produce a flat-text report based on the event data collected in the unformatted event table, for example:

```
java db2evmonfmt -d db_name -ue table_name -ftext -u user_id -p password
```

8. Analyze the report.

Example

The following is an example of a report obtained by using the **db2evmonfmt** Java-based report tool to convert data in the unformatted event table collected by the package cache event monitor:

```
------------------------------------------------------
Event ID              : 1
Event Type            : PKGCACHEBASE
Event Timestamp       : 2009-11-06-12.32.06.442020
Member                : 0
Release               : 9070100
------------------------------------------------------

Package Cache Details
---------------------
Section Type                          : D
Insert Timestamp                      : 2009-11-06-12.28.04.246930
Executable ID                         : 0100000000000000010000000000000000000000002002009110612280424693 2
Package Schema                        :
Package Name                          :
Package Version ID                    :
Section Number                        :
Effective Isolation                   : CS
Number Of Executions                  : 1
Number Of Executions With Metrics     : 1
Prep Time                             : 9
Last Metrics Update                   : 2009-11-06-12.28.07.905942
Executions By Coordinator             : 1
Executions By Coordinator With Metrics : 1
Statement Type                        : DDL, (not Set Constraints)
Query Cost Estimate                   : 1
Statement Package Cache ID            : 1151051235329
Statement Text                        : create event monitor cachestmtevmon for package cache write to unformatted event table
Compilation Environment               : 47454E5F434D504C010000000E000000800100000000000000000000000000000100000004000000000100000
                                        0000000020000000400000008010000000000003000000080000001001000000000B0FA040000000100000018
                                        010000B804B0FA0500000001000000200100000B8040000060000000100000028010000B80400000070000000 10
                                        0000030010000B8040000080000000100000038010000B8040000090000000500000040010000B80400000A00
                                        0000080000004801000B8040000B0000000B0000005001000B8040000100000000A0000006001000B8040
                                        0001300000004000000700100000000000000F0000000400000078010000000000001000000000000000500000
                                        0000000000000000000000000300000000000000030000000000000004E0000000000000004E0000000000000
                                        04E000000000000003120202020000000202044444444444400000000000000000000C00000000020091106
                                        1228040000000000000000000000000000000000000100000000000000
Section Environment                   :

Metrics
-------------------
WLM_QUEUE_TIME_TOTAL                : 0
WLM_QUEUE_ASSIGNMENTS_TOTAL         : 0
FCM_TQ_RECV_WAIT_TIME               : 0
FCM_MESSAGE_RECV_WAIT_TIME          : 0
FCM_TQ_SEND_WAIT_TIME               : 0
FCM_MESSAGE_SEND_WAIT_TIME          : 0
LOCK_WAIT_TIME                      : 0
LOCK_WAITS                          : 0
DIRECT_READ_TIME                    : 0
DIRECT_READ_REQS                    : 0
DIRECT_WRITE_TIME                   : 3
DIRECT_WRITE_REQS                   : 1
LOG_BUFFER_WAIT_TIME                : 0
NUM_LOG_BUFFER_FULL                 : 0
LOG_DISK_WAIT_TIME                  : 0
LOG_DISK_WAITS_TOTAL                : 0
POOL_WRITE_TIME                     : 0
POOL_READ_TIME                      : 33
AUDIT_FILE_WRITE_WAIT_TIME          : 0
AUDIT_FILE_WRITES_TOTAL             : 0
AUDIT_SUBSYSTEM_WAIT_TIME           : 0
AUDIT_SUBSYSTEM_WAITS_TOTAL         : 0
DIAGLOG_WRITE_WAIT_TIME             : 0
DIAGLOG_WRITES_TOTAL                : 0
FCM_SEND_WAIT_TIME                  : 0
FCM_RECV_WAIT_TIME                  : 0
TOTAL_ACT_WAIT_TIME                 : 36
TOTAL_SECTION_SORT_PROC_TIME        : 0
TOTAL_SECTION_SORTS                 : 0
TOTAL_SECTION_SORT_TIME             : 0
TOTAL_ACT_TIME                      : 37
TOTAL_ROUTINE_TIME                  : 0
STMT_EXEC_TIME                      : 3658
COORD_STMT_EXEC_TIME                : 3658
TOTAL_ROUTINE_NON_SECTION_PROC_TIME : 0
```

```
TOTAL_ROUTINE_NON_SECTION_TIME      : 0
TOTAL_SECTION_PROC_TIME             : 1
TOTAL_SECTION_TIME                  : 37
TOTAL_ROUTINE_USER_CODE_PROC_TIME   : 0
TOTAL_ROUTINE_USER_CODE_TIME        : 0
ROWS_READ                           : 19
ROWS_MODIFIED                       : 3
POOL_DATA_L_READS                   : 42
POOL_INDEX_L_READS                  : 83
POOL_TEMP_DATA_L_READS              : 0
POOL_TEMP_INDEX_L_READS             : 0
POOL_XDA_L_READS                    : 0
POOL_TEMP_XDA_L_READS               : 0
TOTAL_CPU_TIME                      : 2243
POOL_DATA_P_READS                   : 13
POOL_TEMP_DATA_P_READS              : 0
POOL_XDA_P_READS                    : 0
POOL_TEMP_XDA_P_READS               : 0
POOL_INDEX_P_READS                  : 33
POOL_TEMP_INDEX_P_READS             : 0
POOL_DATA_WRITES                    : 0
POOL_XDA_WRITES                     : 0
POOL_INDEX_WRITES                   : 0
DIRECT_READS                        : 0
DIRECT_WRITES                       : 2
ROWS_RETURNED                       : 0
DEADLOCKS                           : 0
LOCK_TIMEOUTS                       : 0
LOCK_ESCALS                         : 0
FCM_SENDS_TOTAL                     : 0
FCM_RECVS_TOTAL                     : 0
FCM_SEND_VOLUME                     : 0
FCM_RECV_VOLUME                     : 0
FCM_MESSAGE_SENDS_TOTAL             : 0
FCM_MESSAGE_RECVS_TOTAL             : 0
FCM_MESSAGE_SEND_VOLUME             : 0
FCM_MESSAGE_RECV_VOLUME             : 0
FCM_TQ_SENDS_TOTAL                  : 0
FCM_TQ_RECVS_TOTAL                  : 0
FCM_TQ_SEND_VOLUME                  : 0
FCM_TQ_RECV_VOLUME                  : 0
TQ_TOT_SEND_SPILLS                  : 0
POST_THRESHOLD_SORTS                : 0
POST_SHRTHRESHOLD_SORTS             : 0
SORT_OVERFLOWS                      : 0
AUDIT_EVENTS_TOTAL                  : 0
TOTAL_SORTS                         : 0
THRESH_VIOLATIONS                   : 0
NUM_LW_THRESH_EXCEEDED              : 0
TOTAL_ROUTINE_INVOCATIONS           : 0
```

Using package cache information to identify statements that are candidates for performance tuning:

You can use the package cache event monitor along with in-memory metrics to identify which statements from the package cache are costly to run. Once you know which statements take a long time to run, you can do performance tuning on them.

Before you begin

- You must have a table space in which to store event monitor output before you attempt to create the event monitor. A table space with a page size of at least 8k to store the unformatted event (UE) table produced by the event monitor is recommended. However, even with an 8k page size, the BLOB column used for storing the unformatted event data might not be stored inline within the table. If you want improved performance, consider using a table space with a large page size, such as 32k.
- For partitioned database environments, the table space must extend across all partitions.
- Unless a table space is explicitly named in the CREATE EVENT MONITOR command, the default table space for the database is used.

About this task

This task shows how you can examine all work done on the system between two points in time to find the costliest statements in terms of total CPU time. Using the package cache event monitor together with package cache information reflected in in-memory monitor elements (as returned by the MON_GET_PKG_CACHE_STMT or MON_GET_PKG_CACHE_STMT_DETAILS table functions) is useful because you can see both statements in the cache as well as statements that have been evicted from the cache. Once the costly statements have been identified, you can then do performance tuning on these statements.

Note: You can choose from a number of monitor elements to use when determining which statements are costly to run. In this example, CPU time is used ("total_cpu_time - Total CPU time monitor element" on page 1001). This measurement shows actual CPU resources consumed; it does not reflect things like lock wait time or other time spent during statement execution. You might instead choose to use statement execution time ("stmt_exec_time - Statement execution time monitor element" on page 931), which includes the time spent by all agents in the section, and includes wait times, among other things. You can also choose from many of the other time-spent elements returned by the package cache event monitor. See "Information written to relational tables for a package cache event monitor" on page 87 or "Information written to XML for a package cache event monitor" on page 81 for more information about which monitor elements you can choose from.

Restrictions

In this particular example, the length of the analyzed statements is limited to 3000 characters. This limitation is due to the use of the GROUP BY clause used in the statement, which cannot be used with LOB values, such as the **stmt_text** monitor element.

Procedure

1. Create a package cache event monitor to capture statements as they are removed (evicted) from the package cache. For example, to create an event monitor called EXPENSIVESTMTS, you could use the following SQL:

   ```
   CREATE EVENT MONITOR EXPENSIVESTMTS FOR PACKAGE CACHE WRITE TO UNFORMATTED EVENT TABLE
   ```

   This statement creates a package cache event monitor that writes to a UE table with the same name as the event monitor, EXPENSIVESTMTS, in the default table space for the database. You can override the default name for the UE table using the `TABLE` *table-name* clause. You can also override the table space used for the UE table by using the **IN** *tablespace-name* clause.

   By default, all statements evicted from the package cache are captured by the package cache event monitor. To limit the amount of information collected, you can specify options as part of the CREATE EVENT MONITOR statement that restrict the information collected. Refer to the documentation for the CREATE EVENT MONITOR (package cache) statement for more information.
2. Next, activate the event monitor:

   ```
   SET EVENT MONITOR EXPENSIVESTMTS STATE 1
   ```

   Note: By default, this event monitor starts automatically upon database activation, because the `AUTOSTART` option is applied by default. However, because this event monitor is being created in an already-active database, you must use the **SET EVENT MONITOR** command to start it manually.

3. Connect to the database and run whichever statements, workload or applications for which you are interested in doing performance analysis. You can collect as much information as you like. However, this type of performance tuning works best when you have applications or workloads that run on a regular basis; otherwise adjustments you make for previously executed statements might not have any impact on statements that run in the future.
4. When you are finished collecting data, deactivate the event monitor:

   ```
   SET EVENT MONITOR EXPENSIVESTMTS STATE 0
   ```

5. Extract the data from the UE table that was populated by the event monitor using the EVMON_FORMAT_UE_TO_TABLES procedure.

   ```
   CALL EVMON_FORMAT_UE_TO_TABLES ('PKGCACHE', NULL, NULL, NULL, NULL, NULL, NULL, -1,
        'SELECT * FROM EXPENSIVESTMTS')
   ```

   This procedure examines the UE table TRACKSTMTS produced by the event monitor. It selects all of the records from the UE table, and from them, creates two relational tables from the data collected by the package cache event monitor:

   - PKGCACHE_EVENT
   - PCKCACHE_METRICS

   The first table contains the most frequently used monitor elements and metrics associated with each event captured. The second contains detailed metrics for each event.

   Note: The values in the columns of PKGCACHE_METRICS can also be found in the XML document contained in the METRICS column of the PKGCACHE_EVENT table. They are provided in the PKGCACHE_METRICS table for more convenient, column-oriented access.
6. Query the output from the event monitor to determine which statements took the longest time to run. In this example, total CPU time ("total_cpu_time - Total CPU time monitor element" on page 1001) is the time-spent monitor element used to determine overall cost:

```
WITH STMTS AS
     (
1    SELECT SUM(TOTAL_CPU_TIME) AS TOTAL_CPU_TIME, EXECUTABLE_ID, VARCHAR(STMT_TEXT, 3000) AS STMT_TEXT
     FROM TABLE(MON_GET_PKG_CACHE_STMT(NULL,NULL,NULL,-2)) AS T
     GROUP BY EXECUTABLE_ID, VARCHAR(STMT_TEXT, 3000)
     UNION ALL
2    SELECT SUM(TOTAL_CPU_TIME) AS TOTAL_CPU_TIME, EXECUTABLE_ID, VARCHAR(STMT_TEXT, 3000) AS STMT_TEXT
     FROM PKGCACHE_EVENT E, PKGCACHE_METRICS M WHERE E.XMLID = M.XMLID
     GROUP BY EXECUTABLE_ID, VARCHAR(STMT_TEXT, 3000)
     )
SELECT SUM(TOTAL_CPU_TIME) AS TOTAL_CPU_TIME, STMT_TEXT
FROM STMTS
GROUP BY EXECUTABLE_ID, STMT_TEXT
ORDER BY TOTAL_EXEC_TIME DESC
FETCH FIRST 10 ROWS ONLY;
```

In the preceding example, both the data returned from by the MON_GET_PKG_CACHE_STMT table function (see **1**) and the package cache event monitor (see **2**) are retrieved. Looking at both data sets lets you see data for statements that still exist in the package cache, as well as data for statements that have been evicted from the package cache. Doing so assures that when you evaluate which statements are costly to run that all the statements run between two points in time are considered. The preceding query returns the following results:

Note: For the purposes of printing, the font size of the characters that comprise the sample output that follows has been reduced. This output might be easier

to read from the online version of the topic ("Using package cache information to identify statements that are candidates for performance tuning") in the DB2 Information Center.

```
      TOTAL_CPU_TIME       STMT_TEXT
------------------ ------------------------------------------------------------------------------------------------------------------------
          97796875 select xmlparse(document details_xml) from WLSTATS_TEST3 AS STATS, TABLE(MON_FORMAT_XML_WAIT_TIMES_BY_ROW(STATS.DETAILS_XML)) as t
          94234375 select xmlparse(document details_xml) from WLSTATS_TEST3 AS STATS, TABLE(MON_FORMAT_XML_WAIT_TIMES_BY_ROW(STATS.DETAILS_XML)) as t
          32765625 select xmlparse(document details_xml) from WLSTATS_TEST3 AS STATS, TABLE(MON_FORMAT_XML_WAIT_TIMES_BY_ROW(STATS.DETAILS_XML)) as t
           8375000 select xmlparse(document metrics) from PKGCACHE_EVENT AS EVENTS, table(MON_FORMAT_XML_TIMES_BY_ROW(EVENTS.METRICS)) as t
           1953125 SELECT SUBSTR(METRICS.METRIC_NAME,1,25) AS METRIC_NAME, TOTAL_TIME_VALUE FROM WLSTATS_TESTSTAT AS STATS, TABLE(MON_FORMAT_XML_WAIT
           1771865 CALL XDB_DECOMP_XML_FROM_QUERY (?,?,?,0,0,0,NULL,NULL,1,?,?,?)
           1625000 WITH STMTS AS ( SELECT SUM(TOTAL_CPU_TIME) AS TOTAL_CPU_TIME, EXECUTABLE_ID, VARCHAR(STMT_TEXT, 3000) AS STMT_TEXT FROM TABLE(MON_
            890625 SELECT SUBSTR(METRICS.METRIC_NAME,1,25) AS METRIC_NAME, TOTAL_TIME_VALUE FROM WLSTATS_TEST3 AS DBSTATS, TABLE(MON_FORMAT_XML_TIMES
            843750 INSERT INTO "ASRISK  "."SYSSECTION" SELECT * FROM "SYSIBM"."SYSSECTION"
            671875 SELECT SUBSTR(METRICS.METRIC_NAME,1,25) AS METRIC_NAME, (TOTAL_TIME_VALUE) FROM WLSTATS_TESTSTAT AS STATS, TABLE(MON_FORMAT_XML_WA

  10 record(s) selected
```

Note: The STMT_TEXT column has been truncated for presentation purposes.

What to do next

Use the output from the query shown in step 6 on page 75 to determine which statements to tune.

Using package cache information and db2advis to look for performance improvement opportunities:

The DB2 Design Advisor can analyze SQL statements to make recommendations for how to improve database performance. You can use statements from the package cache (including statements captured by the package cache event monitor) as input to the Design Advisor to identify changes you can make to improve the performance for a given workload, or even for all statements run between two points in time.

Before you begin

- You must have a table space in which to store event monitor output before you attempt to create the event monitor. A table space with a page size of at least 8k to store the unformatted event (UE) table produced by the event monitor is recommended. However, even with an 8k page size, the BLOB column used for storing the unformatted event data might not be stored inline within the table. If you want improved performance, consider using a table space with a large page size, such as 32k.
- For partitioned database environments, the table space must extend across all partitions.
- Unless a table space is explicitly named in the CREATE EVENT MONITOR command, the default table space for the database is used.
- You must have created the explain tables required by the Design Advisor.

About this task

This task shows how you can use the package cache event monitor to track all work done on the system between two points in time, and then use the **db2advis** command to analyze high-cost statements that were run during that period. The output of the db2advis command suggests adjustments or changes you can make to your database to improve its performance, based on the statements run while the package cache event monitor was active. Using the package cache event monitor to capture these statements is useful if the statements in question are no longer in the package cache.

Restrictions

In this particular example, the length of the analyzed statements is limited to 3000 characters. This limitation is due to the use of the GROUP BY clause used in the statement, which cannot be used with LOB values, such as the **stmt_text** monitor element.

Procedure

1. Create a package cache event monitor to capture statements as they are removed (evicted) from the package cache. For example, to create an event monitor called TRACKSTMTS, you could use the following SQL:

   ```
   CREATE EVENT MONITOR TRACKSTMTS FOR PACKAGE CACHE WRITE TO UNFORMATTED EVENT TABLE
   ```

   This statement creates a package cache event monitor that writes to a UE table with the same name as the event monitor, TRACKSTMTS.
2. Next, activate the event monitor:

   ```
   SET EVENT MONITOR TRACKSTMTS STATE 1
   ```
3. Connect to the database and run whichever statements, workload or applications for which you are interested in doing performance analysis. You can collect as much information as you like. However, this type of performance tuning works best when you have applications or workloads that run on a regular basis; otherwise adjustments you make for previously executed statements might not have any impact on statements that run in the future.
4. When you are finished collecting data, deactivate the event monitor:

   ```
   SET EVENT MONITOR TRACKSTMTS STATE 0
   ```
5. Extract the data from the UE table that was populated by the event monitor using the EVMON_FORMAT_UE_TO_TABLES procedure.

   ```
   CALL EVMON_FORMAT_UE_TO_TABLES
       ('PKGCACHE', NULL, NULL, NULL, NULL, NULL, NULL, -1,
        'SELECT * FROM TRACKSTMTS')
   ```

   This procedure creates two relational tables from the data collected by the package cache event monitor:

   - PKGCACHE_EVENT
   - PCKCACHE_METRICS

   The first table contains the most frequently used monitor elements and metrics associated with each event captured. The second contains detailed metrics for each event.

   Note: The values in the columns of PKGCACHE_METRICS can also be found in the XML document contained in the METRICS column of the PKGCACHE_EVENT table. They are provided in the PKGCACHE_METRICS table for more convenient, column-oriented access.
6. Query the output from the event monitor to determine which statements took the longest time to run. In this example, statement execution time ("stmt_exec_time - Statement execution time monitor element" on page 931) is the time-spent monitor element used to determine overall cost. This monitor element is summed across all database partitions.

   Tip: Save the output from the query into a text file. You will us this file in the next step.

```
WITH STMTS AS
(
  SELECT SUM(TOTAL_STMT_EXEC_TIME)/SUM(TOTAL_NUM_COORD_EXEC_WITH_METRICS) AS AVG_TIME_PER_EXEC,
         STMT_TEXT, SUM(NUM_EXECUTIONS) AS NUM_EXECUTIONS, STMT_TYPE_ID
  FROM (
        (
         SELECT    SUM(STMT_EXEC_TIME) AS TOTAL_STMT_EXEC_TIME,
```

```
                SUM(NUM_COORD_EXEC_WITH_METRICS) AS TOTAL_NUM_COORD_EXEC_WITH_METRICS,
                SUM(NUM_COORD_EXEC) AS NUM_EXECUTIONS,
                VARCHAR(STMT_TEXT, 3000) AS STMT_TEXT,
                STMT_TYPE_ID
         FROM      PKGCACHE_EVENT AS E, PKGCACHE_METRICS AS M
         WHERE     E.XMLID = M.XMLID
          AND      NUM_COORD_EXEC_WITH_METRICS > 0
         GROUP BY  VARCHAR(STMT_TEXT, 3000),STMT_TYPE_ID
         ORDER BY TOTAL_NUM_COORD_EXEC_WITH_METRICS DESC
         FETCH FIRST 50 ROWS ONLY
        )
        UNION ALL
        (
         SELECT    SUM(STMT_EXEC_TIME) AS TOTAL_STMT_EXEC_TIME,
                SUM(NUM_COORD_EXEC_WITH_METRICS) AS TOTAL_NUM_COORD_EXEC_WITH_METRICS,
                SUM(NUM_COORD_EXEC) AS NUM_EXECUTIONS,
                VARCHAR(STMT_TEXT, 3000) AS STMT_TEXT,
                STMT_TYPE_ID
         FROM      TABLE(MON_GET_PKG_CACHE_STMT(NULL,NULL,NULL,-2)) AS T
         WHERE     NUM_COORD_EXEC_WITH_METRICS > 0
         GROUP BY  VARCHAR(STMT_TEXT, 3000),STMT_TYPE_ID
         ORDER BY TOTAL_NUM_COORD_EXEC_WITH_METRICS DESC
         FETCH FIRST 50 ROWS ONLY
        )
       ) AS Q_UA
  GROUP BY STMT_TEXT, STMT_TYPE_ID
)
SELECT      '--# SET FREQUENCY ' || NUM_EXECUTIONS || X'0A' || STMT_TEXT || ';'
FROM        STMTS  WHERE STMT_TYPE_ID LIKE 'DML, Select%' OR STMT_TYPE_ID LIKE 'DML, Insert%' 1
ORDER BY    AVG_TIME_PER_EXEC DESC
FETCH FIRST 50 ROWS ONLY;
```

In the preceding sample statement, both the data from the package cache event monitor and the in-memory information from the MON_GET_PKG_CACHE_STMT table function are retrieved. Looking at both data sets lets you see data for statements evicted from the package cache, as well as statements that still exist in the package cache. Doing so assures that when you evaluate which statements are costly to run that you also include statements not yet evicted from the cache. In each case, the query retrieves the top 50 statements from both the active package cache, and the package cache event monitor, based on the number of times the statements ran. Then, from these statements, the top 50 SELECT or INSERT statements are chosen 1 based on the average length of time the statements ran for.

Note: You can choose from a number of monitor elements to use when determining which statements are costly to run. In this example, statement execution time is used. This measurement includes shows the amount of time spent in execution by all members and agents executing this section, and includes things like wait time. You might instead choose to use CPU time (“total_cpu_time - Total CPU time monitor element” on page 1001), which reports only the time spent by the CPU processing the statement. You could also choose from many of the other time-spent elements returned by the package cache event monitor. See “Information written to relational tables for a package cache event monitor” on page 87 or “Information written to XML for a package cache event monitor” on page 81 for more information about which monitor elements you can choose from.

In addition, the query presents the output in the `--# SET FREQUENCY` format the Design Advisor uses for its analysis. The preceding query returns results like the ones that follow:

```
--------------------------------------------------------------------------------------------------------------------------------
--# SET FREQUENCY 1
WITH STMTS AS ( SELECT SUM(TOTAL_STMT_EXEC_TIME)/SUM(TOTAL_NUM_COORD_EXEC_WITH_METRICS) AS AVG_TIME_PER_EXEC, STMT
--# SET FREQUENCY 2
WITH STMTS AS ( SELECT SUM(TOTAL_CPU_TIME) AS TOTAL_CPU_TIME, EXECUTABLE_ID, VARCHAR(STMT_TEXT, 3000) AS STMT_TEXT
--# SET FREQUENCY 1055
SELECT POLICY FROM SYSTOOLS.POLICY WHERE MED='DB2CommonMED' AND DECISION='NOP' AND NAME='CommonPolicy';
--# SET FREQUENCY 99
SELECT CREATOR, NAME, CTIME FROM SYSIBM.SYSTABLES WHERE TYPE='T' OR TYPE='S' OR TYPE='N' WITH UR;
--# SET FREQUENCY 1
```

```
UPDATE SYSTOOLS.HMON_ATM_INFO SET STATS_LOCK = 'N', REORG_LOCK = 'N';
--# SET FREQUENCY 1
UPDATE SYSTOOLS.HMON_ATM_INFO AS ATM SET STATS_FLAG = 'N', REORG_FLAG = 'N' WHERE (ATM.SCHEMA, ATM.NAME) IN   (SEL
--# SET FREQUENCY 1
SELECT POLICY FROM SYSTOOLS.POLICY WHERE MED='DB2TableMaintenanceMED' AND DECISION='TableRunstatsDecision' AND NAM
--# SET FREQUENCY 83
WITH JTAB(JSCHEMA,JNAME) AS (VALUES(TABLE_SCHEMA(CAST(? AS varchar(128)), CAST(? AS varchar(128))), TABLE_NAME (CA
--# SET FREQUENCY 122
WITH VTYPED (NAME, SCHEMA) AS (VALUES(TABLE_NAME (CAST(? AS varchar(128)), CAST(? AS varchar(128))), TABLE_SCHEMA(
--# SET FREQUENCY 1210
SELECT COLNAME, TYPENAME FROM  SYSCAT.COLUMNS WHERE TABNAME='POLICY' AND TABSCHEMA='SYSTOOLS';
--# SET FREQUENCY 105
SELECT TABNAME FROM SYSCAT.TABLES  WHERE TABNAME='HMON_ATM_INFO' AND  TABSCHEMA='SYSTOOLS';
--# SET FREQUENCY 104
DELETE FROM SYSTOOLS.HMON_ATM_INFO AS ATM WHERE NOT EXISTS ( SELECT * FROM SYSIBM.SYSTABLES AS IBM WHERE ATM.NAME
--# SET FREQUENCY 1118
VALUES(SUBSTR(:H00003      ,:H00014,:H00015 ))         INTO :H00009:H00017    ;
--# SET FREQUENCY 274
INSERT INTO "ASRISK"."PKGCACHE_EVENT"("EVENT_ID","XMLID","EVENT_TYPE","EVENT_TIMESTAMP","MEMBER","SECTION_TYPE","I
--# SET FREQUENCY 1
SELECT IBM.TID, IBM.FID FROM SYSIBM.SYSTABLES AS IBM, SYSTOOLS.HMON_ATM_INFO AS ATM WHERE ATM.STATS_FLAG <> 'Y' AN
--# SET FREQUENCY 115
VALUES(SUBSTR(CAST(? AS CLOB(162)),CAST(? AS INTEGER),CAST(? AS INTEGER)));
--# SET FREQUENCY 8227

⋮

-# SET FREQUENCY 532
SELECT TBNAME, TBCREATOR FROM "ASRISK  ".SYSINDEXES WHERE NAME = 'INDCOLUMNS01' AND CREATOR = 'SYSIBM  ';
--# SET FREQUENCY 105
SELECT TABNAME FROM SYSCAT.TABLES  WHERE TABNAME='HMON_COLLECTION' AND  TABSCHEMA='SYSTOOLS';
--# SET FREQUENCY 4091
SELECT STATS_LOCK, REORG_LOCK FROM SYSTOOLS.HMON_ATM_INFO WHERE SCHEMA = ? AND NAME = ? AND CREATE_TIME = ? FOR UP
--# SET FREQUENCY 17100
SELECT CREATE_TIME FROM SYSTOOLS.HMON_ATM_INFO WHERE SCHEMA = ? AND NAME = ? FOR UPDATE;
--# SET FREQUENCY 524
SELECT COUNT(*) FROM "SYSIBM".SYSTABLES WHERE NAME = 'SYSDATAPARTITIONEXPRESSION' AND CREATOR = 'SYSIBM  ' AND TYP
--# SET FREQUENCY 532
SELECT COUNT(*) FROM "SYSIBM".SYSTABLES WHERE NAME = 'SYSCOLUMNS' AND CREATOR = 'SYSIBM  ' AND TYPE = 'S';

  47 record(s) selected
```

Note: The lines in the preceding sample output have been truncated for presentation purposes.

7. Create an input file for the **db2advis** command using the statements returned by the query in step 6 on page 77. (For more information about creating input files for the **db2advis** command, refer to the reference documentation for that command.)
8. Run the **db2advis** command using the input file created in step 7. For example, if the input file you create is called `pkgcache_stmts.txt`, run a command like the one that follows:

   ```
   db2advis -d customer -i pkgcache_stmts.txt -m MICP
   ```

   where
   - **-d CUSTOMER** identifies the name of the database for which you are getting recommendations
   - **-i pkgcache_stmts.txt** identifies the name of the input file for **db2advis**
   - **-m MICP** is a directive to the db2advis command to produce the following recommendations to improve performance:
     - **M** New materialized query tables
     - **I** New indexes
     - **C** Converting standard tables to multidimensional clustering tables (MQTs)
     - **P** Repartitioning existing indexes

Results

The Design Advisor returns recommendations like ones that follow:

```
execution started at timestamp 2010-03-16-14.25.57.562000
Using the default table space name USERSPACE1
found [47] SQL statements from the input file
excluding statement [0] from the workload.
excluding statement [1] from the workload.
```

```
excluding statement [19] from the workload.
excluding statement [39] from the workload.
Recommending indexes...
Recommending MQTs...
Recommending Multi-Dimensional Clusterings...
Found 19 user defined views in the catalog table
Found [17] candidate MQTs
Getting cost of workload with MQTs
total disk space needed for initial set [   0.159] MB
total disk space constrained to         [  69.215] MB
  2  indexes in current solution
  0  MQTs in current solution
total disk space needed for initial set [   0.024] MB
total disk space constrained to         [ 103.822] MB
No useful Multi-dimensional Clustering dimensions for this workload
 [5651.8281] timerons  (without recommendations)
 [5519.8281] timerons  (with current solution)
 [2.34%] improvement


--
--
-- LIST OF MODIFIED CREATE-TABLE STATEMENTS WITH RECOMMENDED PARTITIONING KEYS AND TABLESPACES AND/OR RECOMMENDED MULTI-DIMENSIONAL CLUSTERINGS
-- ===========================
--  No new partitioning keys or tablespaces are recommended for this workload.


--
--
-- LIST OF RECOMMENDED MQTs
-- ===========================


--
--
-- RECOMMENDED EXISTING MQTs
-- ===========================


--
--
-- UNUSED EXISTING MQTs
-- ===========================
-- DROP TABLE "ASRISK  "."ADEFUSR";


--
--
-- RECOMMENDED CLUSTERING INDEXES
-- ===========================


--
--
-- LIST OF RECOMMENDED INDEXES
-- ===========================
-- index[1],    0.024MB
   CREATE INDEX "ASRISK  "."IDX003161830530000" ON "ASRISK  "."SYSINDEXES"
   ("CREATOR" ASC, "NAME" ASC, "TBCREATOR" ASC, "TBNAME"
   ASC) ALLOW REVERSE SCANS COLLECT SAMPLED DETAILED STATISTICS;
   COMMIT WORK ;


--
--
-- RECOMMENDED EXISTING INDEXES
-- ===========================
-- RUNSTATS ON TABLE "SYSTOOLS"."POLICY" FOR SAMPLED DETAILED INDEX "SYSTOOLS"."POLICY_UNQ" ;
-- COMMIT WORK ;
-- RUNSTATS ON TABLE "SYSTOOLS"."HMON_ATM_INFO" FOR SAMPLED DETAILED INDEX "SYSTOOLS"."ATM_UNIQ" ;
-- COMMIT WORK ;
-- RUNSTATS ON TABLE "SYSIBM  "."SYSDATAPARTITIONS" FOR SAMPLED DETAILED INDEX "SYSIBM  "."INDDATAPARTITIONS03" ;
-- COMMIT WORK ;
-- RUNSTATS ON TABLE "SYSIBM  "."SYSTABLES" FOR SAMPLED DETAILED INDEX "SYSIBM  "."INDTABLES01" ;
-- COMMIT WORK ;
-- RUNSTATS ON TABLE "SYSIBM  "."SYSTABLESPACES" FOR SAMPLED DETAILED INDEX "SYSIBM  "."INDTABLESPACES04" ;
-- COMMIT WORK ;
-- RUNSTATS ON TABLE "SYSIBM  "."SYSCOLUMNS" FOR SAMPLED DETAILED INDEX "SYSIBM  "."INDCOLUMNS01" ;
-- COMMIT WORK ;
-- RUNSTATS ON TABLE "SYSIBM  "."SYSINDEXES" FOR SAMPLED DETAILED INDEX "SYSIBM  "."INDINDEXES02" ;
-- COMMIT WORK ;
-- RUNSTATS ON TABLE "SYSIBM  "."SYSTRIGGERS" FOR SAMPLED DETAILED INDEX "SYSIBM  "."INDTRIGGERS02" ;
-- COMMIT WORK ;


--
--
-- UNUSED EXISTING INDEXES
-- ===========================
-- DROP INDEX "ASRISK  "."PKGCACHE_EVENT_IND1";
```

```
-- ==========================
--

-- ====ADVISOR DETAILED XML OUTPUT=============
-- ==(Benefits do not include clustering recommendations)==

⋮
```

Note: The output from the Design Advisor has been truncated for presentation purposes.

What to do next

Use the output from the Design Advisor to help when deciding what changes to make to your database to improve performance.

Information written to XML for a package cache event monitor

Information written for a package cache event monitor from the EVMON_FORMAT_UE_TO_XML table function. This is also documented in the `DB2EvmonPkgCache.xsd` file.

db2_pkgcache_event

The main schema that describes a package cache event in details.

Element content: (section_type , insert_timestamp , executable_id , package_schema , package_name , package_version_id , section_number {zero or one times (?)} , effective_isolation , num_executions , num_exec_with_metrics , prep_time , last_metrics_update , num_coord_exec , num_coord_exec_with_metrics , stmt_type_id , query_cost_estimate , stmt_pkg_cache_id , stmt_text , comp_env_desc , section_env , activity_metrics , ANY content (skip) {zero or more (*)})

Attributes:

QName	Type	Fixed	Default	Use	Annotation
id	xs:long			required	
type	xs:string - Max length: 32 (PKGCACHE_BASE, PKGCACHE_DETAILED)			required	
timestamp	xs:dateTime			required	
member	member_type			required	
release	xs:long			required	
ANY attribute from ANY namespace					

section_type

The type of SQL statement processed. Possible values: D:Dynamic or S:Static. See monitor element “section_type - Section type indicator monitor element” on page 902 for more details.

Contained by: db2_pkgcache_event

Element content:

Type	Facet
xs:string	Max length: 1

insert_timestamp

The time when the variation or section was inserted into the cache. See monitor element “insert_timestamp - Insert timestamp monitor element” on page 696 for more details.

Contained by: db2_pkgcache_event

Element content:

Type	Facet
xs:dateTime	

executable_id

A binary token generated on the data server that uniquely identifies the SQL statement section that was executed. See monitor element “executable_id - Executable ID monitor element” on page 639 for more details.

Contained by: db2_pkgcache_event

Element content:

Type	Facet
xs:hexBinary	Max length: 32

package_schema

The schema name of the package associated with an SQL statement. See monitor element “package_schema - Package schema monitor element” on page 792 for more details.

Contained by: db2_pkgcache_event

Element content:

Type	Facet
xs:string	Max length: 128

package_name

The name of the package that contains the SQL statement currently executing. See monitor element “package_name - Package name monitor element” on page 791 for more details.

Contained by: db2_pkgcache_event

Element content:

Type	Facet
xs:string	Max length: 128

package_version_id

The package version identifies the version identifier of the package that contains the SQL statement currently executing. See monitor element “package_version_id - Package version monitor element” on page 792 for more details.

Contained by: db2_pkgcache_event

Element content:

Type	Facet
xs:string	Max length: 64

section_number

The internal section number in the package for the SQL statement currently processing or most recently processed. See monitor element “section_number - Section number monitor element” on page 901 for more details.

Contained by: db2_pkgcache_event

Element content:

Type	Facet
xs:long	

effective_isolation

The isolation value in effect for the SQL statement while it was being run. See monitor element “effective_isolation - Effective isolation monitor element” on page 633 for more details.

Contained by: db2_pkgcache_event

Element content:

Type	Facet
xs:string	Max length: 2

Attributes:

QName	Type	Fixed	Default	Use	Annotation
id	xs:long			optional	

num_executions

The number times the SQL statement has been executed. See monitor element "num_executions - Statement executions monitor element" on page 773 for more details.

Contained by: db2_pkgcache_event

Element content:

Type	Facet
xs:long	

num_exec_with_metrics

The number times the SQL statement has been executed with the metrics collected. See monitor element "num_exec_with_metrics - Number of executions with metrics collected monitor element" on page 773 for more details.

Contained by: db2_pkgcache_event

Element content:

Type	Facet
xs:long	

prep_time

Time in milliseconds required to prepare an SQL statement if the activity is an SQL statement. See monitor element "prep_time - Preparation time monitor element" on page 860 for more details.

Contained by: db2_pkgcache_event

Element content:

Type	Facet
xs:long	

last_metrics_update

Timestamp reflecting the last time metrics were updated for this cache entry. See monitor element "last_metrics_update - Metrics last update timestamp monitor element" on page 712 for more details.

Contained by: db2_pkgcache_event

Element content:

Type	Facet
xs:dateTime	

num_coord_exec

The number of times this section was executed by a coordinator agent. See monitor element num_coord_exec for more details.

Contained by: db2_pkgcache_event

Element content:

Type	Facet
xs:long	

num_coord_exec_with_metrics

The number of times this section was executed by a coordinator agent and monitoring metrics were being captured. See monitor element "num_coord_exec_with_metrics - Number of executions by coordinator agent with metrics monitor element" on page 772 for more details.

Contained by: db2_pkgcache_event

Element content:

Type	Facet
xs:long	

stmt_type_id

Statement type identifier. See monitor element "stmt_type_id - Statement type identifier monitor element" on page 942 for more details.

Contained by: db2_pkgcache_event

Element content:

Type	Facet
xs:string	Max length: 32

Attributes:

QName	Type	Fixed	Default	Use	Annotation
id	xs:long			optional	

query_cost_estimate

Estimated cost for a query, as determined by the SQL compiler. See monitor element "query_cost_estimate - Query cost estimate monitor element" on page 869 for more details.

Contained by: db2_pkgcache_event

Element content:

Type	Facet
xs:long	

stmt_pkg_cache_id

See monitor element "stmt_pkgcache_id - Statement package cache identifier monitor element" on page 937 for more details.

Contained by: db2_pkgcache_event

Element content:

Type	Facet
xs:long	

stmt_text

The text of the SQL statement. See monitor element "stmt_text - SQL statement text monitor element" on page 940 for more details.

Contained by: db2_pkgcache_event

Element content:

Type	Facet
xs:string	Max length: 2097152

comp_env_desc

"comp_env_desc - Compilation environment monitor element" on page 565

Contained by: db2_pkgcache_event

Element content:

Type	Facet
xs:hexBinary	Max length: 10240

section_env

A BLOB that contains the section for an SQL statement. See monitor element "section_env - Section environment monitor element" on page 901 for more details.

Contained by: db2_pkgcache_event

Element content:

Type	Facet
xs:hexBinary	Max length: 157286400

activity_metrics

The activity metrics for this cache entry.

Contained by: db2_pkgcache_event

Element content: (wlm_queue_time_total , wlm_queue_assignments_total , fcm_tq_recv_wait_time , fcm_message_recv_wait_time , fcm_tq_send_wait_time , fcm_message_send_wait_time , lock_wait_time , lock_waits , direct_read_time , direct_read_reqs , direct_write_time , direct_write_reqs , log_buffer_wait_time , num_log_buffer_full , log_disk_wait_time , log_disk_waits_total , pool_write_time , pool_read_time , audit_file_write_wait_time , audit_file_writes_total , audit_subsystem_wait_time , audit_subsystem_waits_total , diaglog_write_wait_time , diaglog_writes_total , fcm_send_wait_time , fcm_recv_wait_time , total_act_wait_time , total_section_sort_proc_time , total_section_sort_time , total_section_sorts , total_act_time , rows_read , rows_modified , pool_data_l_reads , pool_index_l_reads , pool_temp_data_l_reads , pool_temp_index_l_reads , pool_xda_l_reads , pool_temp_xda_l_reads , total_cpu_time , pool_data_p_reads , pool_temp_data_p_reads , pool_xda_p_reads , pool_temp_xda_p_reads , pool_index_p_reads , pool_temp_index_p_reads , pool_data_writes , pool_xda_writes , pool_index_writes , direct_reads , direct_writes , rows_returned , deadlocks , lock_timeouts , lock_escals , fcm_sends_total , fcm_recvs_total , fcm_send_volume , fcm_recv_volume , fcm_message_sends_total , fcm_message_recvs_total , fcm_message_send_volume , fcm_message_recv_volume , fcm_tq_sends_total , fcm_tq_recvs_total , fcm_tq_send_volume , fcm_tq_recv_volume , tq_tot_send_spills , post_threshold_sorts , post_shrthreshold_sorts , sort_overflows , audit_events_total , total_sorts , stmt_exec_time , coord_stmt_exec_time {zero or one times (?)} , total_routine_non_sect_proc_time , total_routine_non_sect_time , total_section_proc_time , total_section_time , total_app_section_executions , total_routine_user_code_proc_time , total_routine_user_code_time , total_routine_time , thresh_violations , num_lw_thresh_exceeded , total_routine_invocations , ANY content (skip) {zero or more (*)})

Attributes:

QName	Type	Fixed	Default	Use	Annotation
release	xs:long			required	
ANY attribute from ANY namespace					

Information written to relational tables for a package cache event monitor

Information written for a package cache event monitor from the EVMON_FORMAT_UE_TO_TABLES table function. This is also documented in the `DB2EvmonPkgCache.xsd` file.

Table 9. Information returned for a package cache event monitor: Table name: PKGCACHE_EVENT

Column Name	Data Type	Description
XMLID	VARCHAR(1000) NOT NULL	
EVENT_ID	BIGINT NOT NULL	

Table 9. Information returned for a package cache event monitor: Table name: PKGCACHE_EVENT (continued)

Column Name	Data Type	Description
EVENT_TYPE	VARCHAR(128) NOT NULL	
EVENT_TIMESTAMP	TIMESTAMP NOT NULL	
MEMBER	SMALLINT NOT NULL	"member - Database member monitor element" on page 760
SECTION_TYPE	CHAR(1)	"section_type - Section type indicator monitor element" on page 902
INSERT_TIMESTAMP	TIMESTAMP	"insert_timestamp - Insert timestamp monitor element" on page 696
EXECUTABLE_ID	VARCHAR(32) FOR BIT DATA	"executable_id - Executable ID monitor element" on page 639
PACKAGE_SCHEMA	VARCHAR(128)	"package_schema - Package schema monitor element" on page 792
PACKAGE_NAME	VARCHAR(128)	"package_name - Package name monitor element" on page 791
PACKAGE_VERSION_ID	VARCHAR(64)	"package_version_id - Package version monitor element" on page 792
SECTION_NUMBER	BIGINT	"section_number - Section number monitor element" on page 901
EFFECTIVE_ISOLATION	CHAR(2)	"effective_isolation - Effective isolation monitor element" on page 633
NUM_EXECUTIONS	BIGINT	"num_executions - Statement executions monitor element" on page 773
NUM_EXEC_WITH_METRICS	BIGINT	"num_exec_with_metrics - Number of executions with metrics collected monitor element" on page 773
PREP_TIME	BIGINT	"prep_time - Preparation time monitor element" on page 860
LAST_METRICS_UPDATE	TIMESTAMP	"last_metrics_update - Metrics last update timestamp monitor element" on page 712
NUM_COORD_EXEC	BIGINT	"num_coord_exec - Number of executions by coordinator agent monitor element" on page 772
NUM_COORD_EXEC_WITH_METRICS	BIGINT	"num_coord_exec_with_metrics - Number of executions by coordinator agent with metrics monitor element" on page 772
STMT_TYPE_ID	VARCHAR(32)	"stmt_type_id - Statement type identifier monitor element" on page 942
QUERY_COST_ESTIMATE	BIGINT	"query_cost_estimate - Query cost estimate monitor element" on page 869
STMT_PKG_CACHE_ID	BIGINT	
STMT_TEXT	CLOB(2M)	"stmt_text - SQL statement text monitor element" on page 940
COMP_ENV_DESC	BLOB(10K)	"comp_env_desc - Compilation environment monitor element" on page 565

Table 9. Information returned for a package cache event monitor: Table name: PKGCACHE_EVENT (continued)

Column Name	Data Type	Description
METRICS	BLOB(1M)	XML document containing metrics-related monitor elements. The metrics in this document are the same as those described in the PKGCACHE_METRICS table that appears later in this topic. See Chapter 3, "Interfaces that return monitor data in XML documents," on page 9 for more information.
SECTION_ENV	BLOB(150M)	"section_env - Section environment monitor element" on page 901

Table 10. Information returned for a package cache event monitor: Table name: PKGCACHE_METRICS. The metrics in this table are the same as those returned in the METRICS monitor element in the PKGCACHE_EVENT table

Column Name	Data Type	Description
XMLID	VARCHAR(1000) NOT NULL	
TOTAL_ACT_TIME	BIGINT	"total_act_time - Total activity time monitor element" on page 988
TOTAL_ACT_WAIT_TIME	BIGINT	"total_act_wait_time - Total activity wait time monitor element" on page 989
TOTAL_CPU_TIME	BIGINT	"total_cpu_time - Total CPU time monitor element" on page 1001
POOL_READ_TIME	BIGINT	"pool_read_time - Total buffer pool physical read time monitor element" on page 833
POOL_WRITE_TIME	BIGINT	"pool_write_time - Total buffer pool physical write time monitor element" on page 847
DIRECT_READ_TIME	BIGINT	"direct_read_time - Direct read time monitor element" on page 622
DIRECT_WRITE_TIME	BIGINT	"direct_write_time - Direct write time monitor element" on page 627
LOCK_WAIT_TIME	BIGINT	"lock_wait_time - Time waited on locks monitor element" on page 734
TOTAL_SECTION_SORT_TIME	BIGINT	"total_section_sort_time - Total section sort time monitor element" on page 1030
TOTAL_SECTION_SORT_PROC_TIME	BIGINT	"total_section_sort_proc_time - Total section sort processing time monitor element" on page 1028
TOTAL_SECTION_SORTS	BIGINT	"total_section_sorts - Total section sorts monitor element" on page 1032
LOCK_ESCALS	BIGINT	"lock_escals - Number of lock escalations monitor element" on page 721
LOCK_WAITS	BIGINT	"lock_waits - Lock waits monitor element" on page 737
ROWS_MODIFIED	BIGINT	"rows_modified - Rows modified monitor element" on page 890
ROWS_READ	BIGINT	"rows_read - Rows read monitor element" on page 892

Table 10. Information returned for a package cache event monitor: Table name: PKGCACHE_METRICS (continued). The metrics in this table are the same as those returned in the METRICS monitor element in the PKGCACHE_EVENT table

Column Name	Data Type	Description
ROWS_RETURNED	BIGINT	"rows_returned - Rows returned monitor element" on page 893
DIRECT_READS	BIGINT	"direct_reads - Direct reads from database monitor element" on page 624
DIRECT_READ_REQS	BIGINT	"direct_read_reqs - Direct read requests monitor element" on page 620
DIRECT_WRITES	BIGINT	"direct_writes - Direct writes to database monitor element" on page 629
DIRECT_WRITE_REQS	BIGINT	"direct_write_reqs - Direct write requests monitor element" on page 626
POOL_DATA_L_READS	BIGINT	"pool_data_l_reads - Buffer pool data logical reads monitor element" on page 816
POOL_TEMP_DATA_L_READS	BIGINT	"pool_temp_data_l_reads - Buffer pool temporary data logical reads monitor element" on page 835
POOL_XDA_L_READS	BIGINT	"pool_xda_l_reads - Buffer pool XDA data logical reads monitor element" on page 849
POOL_TEMP_XDA_L_READS	BIGINT	"pool_temp_xda_l_reads - Buffer pool temporary XDA data logical reads monitor element" on page 843
POOL_INDEX_L_READS	BIGINT	"pool_index_l_reads - Buffer pool index logical reads monitor element" on page 825
POOL_TEMP_INDEX_L_READS	BIGINT	"pool_temp_index_l_reads - Buffer pool temporary index logical reads monitor element" on page 839
POOL_DATA_P_READS	BIGINT	"pool_data_p_reads - Buffer pool data physical reads monitor element" on page 818
POOL_TEMP_DATA_P_READS	BIGINT	"pool_temp_data_p_reads - Buffer pool temporary data physical reads monitor element" on page 837
POOL_XDA_P_READS	BIGINT	"pool_xda_p_reads - Buffer pool XDA data physical reads monitor element" on page 851
POOL_TEMP_XDA_P_READS	BIGINT	"pool_temp_xda_p_reads - Buffer pool temporary XDA data physical reads monitor element" on page 845
POOL_INDEX_P_READS	BIGINT	"pool_index_p_reads - Buffer pool index physical reads monitor element" on page 827
POOL_TEMP_INDEX_P_READS	BIGINT	"pool_temp_index_p_reads - Buffer pool temporary index physical reads monitor element" on page 841
POOL_DATA_WRITES	BIGINT	"pool_data_writes - Buffer pool data writes monitor element" on page 820

Table 10. Information returned for a package cache event monitor: Table name: PKGCACHE_METRICS (continued). The metrics in this table are the same as those returned in the METRICS monitor element in the PKGCACHE_EVENT table

Column Name	Data Type	Description
POOL_XDA_WRITES	BIGINT	"pool_xda_writes - Buffer pool XDA data writes monitor element" on page 853
POOL_INDEX_WRITES	BIGINT	"pool_index_writes - Buffer pool index writes monitor element" on page 829
TOTAL_SORTS	BIGINT	"total_sorts - Total sorts monitor element" on page 1036
POST_THRESHOLD_SORTS	BIGINT	"post_threshold_sorts - Post threshold sorts monitor element" on page 858
POST_SHRTHRESHOLD_SORTS	BIGINT	"post_shrthreshold_sorts - Post shared threshold sorts monitor element" on page 856
SORT_OVERFLOWS	BIGINT	"sort_overflows - Sort overflows monitor element" on page 915
WLM_QUEUE_TIME_TOTAL	BIGINT	"wlm_queue_time_total - Workload manager total queue time monitor element" on page 1060
WLM_QUEUE_ASSIGNMENTS_TOTAL	BIGINT	"wlm_queue_assignments_total - Workload manager total queue assignments monitor element" on page 1059
DEADLOCKS	BIGINT	"deadlocks - Deadlocks detected monitor element" on page 614
FCM_RECV_VOLUME	BIGINT	"fcm_recv_volume - FCM received volume monitor element" on page 647
FCM_RECVS_TOTAL	BIGINT	"fcm_recvs_total - FCM receives total monitor element" on page 650
FCM_SEND_VOLUME	BIGINT	"fcm_send_volume - FCM send volume monitor element" on page 651
FCM_SENDS_TOTAL	BIGINT	"fcm_sends_total - FCM sends total monitor element" on page 654
FCM_RECV_WAIT_TIME	BIGINT	"fcm_recv_wait_time - FCM received wait time monitor element" on page 648
FCM_SEND_WAIT_TIME	BIGINT	"fcm_send_wait_time - FCM send wait time monitor element" on page 652
LOCK_TIMEOUTS	BIGINT	"lock_timeouts - Number of lock timeouts monitor element" on page 732
LOG_BUFFER_WAIT_TIME	BIGINT	"log_buffer_wait_time - Log buffer wait time monitor element" on page 740
NUM_LOG_BUFFER_FULL	BIGINT	"num_log_buffer_full - Number of times full log buffer caused agents to wait monitor element" on page 775
LOG_DISK_WAIT_TIME	BIGINT	"log_disk_wait_time - Log disk wait time monitor element" on page 741
LOG_DISK_WAITS_TOTAL	BIGINT	"log_disk_waits_total - Total log disk waits monitor element" on page 743

Table 10. Information returned for a package cache event monitor: Table name: PKGCACHE_METRICS (continued). The metrics in this table are the same as those returned in the METRICS monitor element in the PKGCACHE_EVENT table

Column Name	Data Type	Description
TOTAL_ROUTINE_TIME	BIGINT	"total_routine_time - Total routine time monitor element" on page 1018
TOTAL_ROUTINE_INVOCATIONS	BIGINT	"total_routine_invocations - Total routine invocations monitor elements" on page 1016
COORD_STMT_EXEC_TIME	BIGINT	"coord_stmt_exec_time - Execution time for statement by coordinator agent monitor element" on page 591
STMT_EXEC_TIME	BIGINT	"stmt_exec_time - Statement execution time monitor element" on page 931
TOTAL_SECTION_TIME	BIGINT	"total_section_time - Total section time monitor element" on page 1033
TOTAL_SECTION_PROC_TIME	BIGINT	"total_section_proc_time - Total section processing time monitor element" on page 1027
TOTAL_ROUTINE_NON_SECT_TIME	BIGINT	"total_routine_non_sect_time - Non-section routine execution time monitor elements" on page 1017
TOTAL_ROUTINE_NON_SECT_ PROC_TIME	BIGINT	"total_routine_non_sect_proc_time - Non-section processing time monitor element" on page 1017
FCM_TQ_RECV_WAIT_TIME	BIGINT	"fcm_tq_recv_wait_time - FCM table queue received wait time monitor element" on page 656
FCM_MESSAGE_RECV_WAIT_TIME	BIGINT	"fcm_message_recv_wait_time - FCM message received wait time monitor element" on page 641
FCM_TQ_SEND_WAIT_TIME	BIGINT	"fcm_tq_send_wait_time - FCM table queue send wait time monitor element" on page 659
FCM_MESSAGE_SEND_WAIT_TIME	BIGINT	"fcm_message_send_wait_time - FCM message send wait time monitor element" on page 645
AUDIT_FILE_WRITE_WAIT_TIME	BIGINT	"audit_file_write_wait_time - Audit file write wait time monitor element" on page 532
AUDIT_FILE_WRITES_TOTAL	BIGINT	"audit_file_writes_total - Total audit files written monitor element" on page 533
AUDIT_SUBSYSTEM_WAIT_TIME	BIGINT	"audit_subsystem_wait_time - Audit subsystem wait time monitor element" on page 534
AUDIT_SUBSYSTEM_WAITS_TOTAL	BIGINT	"audit_subsystem_waits_total - Total audit subsystem waits monitor element" on page 536
DIAGLOG_WRITE_WAIT_TIME	BIGINT	"diaglog_write_wait_time - Diagnostic log file write wait time monitor element" on page 618

Table 10. Information returned for a package cache event monitor: Table name: PKGCACHE_METRICS (continued). The metrics in this table are the same as those returned in the METRICS monitor element in the PKGCACHE_EVENT table

Column Name	Data Type	Description
DIAGLOG_WRITES_TOTAL	BIGINT	"diaglog_writes_total - Total diagnostic log file writes monitor element" on page 619
FCM_MESSAGE_SENDS_TOTAL	BIGINT	"fcm_message_sends_total - Total FCM message sends monitor element" on page 646
FCM_MESSAGE_RECVS_TOTAL	BIGINT	"fcm_message_recvs_total - Total FCM message receives monitor element" on page 643
FCM_MESSAGE_SEND_VOLUME	BIGINT	"fcm_message_send_volume - FCM message send volume monitor element" on page 644
FCM_MESSAGE_RECV_VOLUME	BIGINT	"fcm_message_recv_volume - FCM message received volume monitor element" on page 640
FCM_TQ_SENDS_TOTAL	BIGINT	"fcm_tq_sends_total - FCM table queue send total monitor element" on page 660
FCM_TQ_RECVS_TOTAL	BIGINT	"fcm_tq_recvs_total - FCM table queue receives total monitor element" on page 657
FCM_TQ_SEND_VOLUME	BIGINT	"fcm_tq_send_volume - FCM table queue send volume monitor element" on page 658
FCM_TQ_RECV_VOLUME	BIGINT	"fcm_tq_recv_volume - FCM table queue received volume monitor element" on page 655
TQ_TOT_SEND_SPILLS	BIGINT	"tq_tot_send_spills - Total number of table queue buffers overflowed monitor element" on page 1044
AUDIT_EVENTS_TOTAL	BIGINT	"audit_events_total - Total audit events monitor element" on page 531
TOTAL_APP_SECTION_EXECUTIONS	BIGINT	"total_app_section_executions - Total application section executions monitor element" on page 993
TOTAL_ROUTINE_USER_CODE_ PROC_TIME	BIGINT	"total_routine_user_code_proc_time - Total routine user code processing time monitor element" on page 1020
TOTAL_ROUTINE_USER_CODE_TIME	BIGINT	"total_routine_user_code_time - Total routine user code time monitor element" on page 1021
THRESH_VIOLATIONS	BIGINT	"thresh_violations - Number of threshold violations monitor element" on page 981
NUM_LW_THRESH_EXCEEDED	BIGINT	"num_lw_thresh_exceeded - Number of lock wait thresholds exceeded monitor element" on page 778

Monitoring unit of work events

The unit of work event monitor records an event whenever a unit of work is completed, that is, whenever there is a commit or a rollback. This historical information about individual units of work is useful for chargeback purposes (charging by CPU usage) and for monitoring compliance with response time service level objectives.

The unit of work event monitor is one way to perform system perspective monitoring with request metrics. The most closely related alternatives or complements to the unit of work event monitor are either the statistics event monitor or the MON_GET_UNIT_OF_WORK and MON_GET_UNIT_OF_WORK_DETAILS table functions.

Starting with DB2 Version 9.7 Fix Pack 1, using the unit of work event monitor, you can collect a listing of packages used within a unit of work, as well as the nesting level at which it was used, to help facilitate stored procedure troubleshooting.

To create the unit of work event monitor and collect unit of work event monitor data, you must have DBADM or SQLADM authority.

Creating a unit of work event monitor and configuring data collection

Before you create a unit of work event monitor, identify the table space where you plan to store the unformatted event table for your event monitor. The recommended practice is to have a table space dedicated and configured to store the unformatted event table associated with any event monitor. Create the unit of work event monitor in a tablespace with at least 8K pagesize to ensure that the event data is contained within the inlined BLOB column of the unformatted event table. If the BLOB column is not inlined, then the performance of writing and reading the events to the unformatted event table might not be efficient.

The database manager attempts to inline the event_data BLOB column in the unformatted event table, but this is not always possible. To check that the rows in the unformatted event table have been inlined, use the ADMIN_IS_INLINED function. If the rows have not been inlined, use the ADMIN_EST_INLINE_LENGTH functions to determine how much space the rows need.

Your other options when you create an event monitor are to specify any existing table space or to not specify any and have one chosen by default.

To setup a unit of work event monitor using defaults and best practices, complete the following steps:

1. Create the event monitor by issuing the CREATE EVENT MONITOR statement. The following example uses defaults where possible and specifies to store the unformatted event table in an existing table space:

   ```
   CREATE EVENT MONITOR MY_UOW_EVMON
     FOR UNIT OF WORK
      WRITE TO UNFORMATTED EVENT TABLE (IN MY_EVMON_TABLESPACE)
   ```

2. Configure what data to collect. The following statement illustrates a simple approach:

   ```
   db2 update db cfg for dbname using mon_uow_data base
   ```

Configuring data collection

To configure data collection, you must also specify the subset of the system workload for which to capture events and how much detail to collect for each event. By default unit of work data is not collected. You can change the default settings by using one of the following settings:

- The **mon_uow_data** database configuration parameter
- The COLLECT UNIT OF WORK DATA clause of the CREATE and ALTER WORKLOAD statements.

The following levels for data collection are available to you:

NONE
No unit of work data collected.

BASE Basic unit of work data collected.

PACKAGE LIST
The package list for transactions associated with this unit of work, as well as the basic unit of work data, is collected.

If either the **mon_uow_data** database configuration parameter or the COLLECT UNIT OF WORK DATA clause of the CREATE/ALTER WORKLOAD statement is set to BASE, then that is the effective setting for the workload.

If you want to enable data collection for only selected workloads, then set **mon_uow_data** database configuration parameter to NONE and set the level to BASE for the desired workloads.

Requests metrics is one of the types of information that you can collect with a unit of work event monitor. The unit of work event monitor is one of the interfaces affected by the setting for request metric collection. By default, request metrics are collected and reported in applicable table functions and event monitors, including the unit of work event monitor. You can change the default setting by using one of the following settings:

- The **mon_req_metrics** database configuration parameter
- The COLLECT REQUEST METRICS clause of the CREATE/ALTER SERVICE CLASS statement for a service superclass.

Changing these settings affects any table function or event monitor that can report request metrics.

Accessing event data captured by a unit of work event monitor

This type of event monitor writes data in a binary format to an unformatted event table. You can access this data using the following table functions:

- EVMON_FORMAT_UE_TO_XML - extracts data from an unformatted event table into an XML document.
- EVMON_FORMAT_UE_TO_TABLES - extracts data from an unformatted event table into a set of relational tables.

Use these table functions to specify the data to extract using a SELECT statement. You have full control over selection, ordering, and other aspects provided by the SELECT statement.

If you are generating package listing information, EVMON_FORMAT_UE_TO_XML generates a single XML document that contains both the basic unit of work event monitor data as well as the package listing. EVMON_FORMAT_UE_TO_TABLES produces two tables, one for the base unit of work event monitor information, and another for the package listing information. You can join the two using the values contained in the MEMBER, APPLICATION_ID and UOW_ID columns.

You can also use the **db2evmonfmt** command to perform the following tasks:

- Select events of interest based on the following attributes: event ID, event type, time period, application, workload, or service class.
- Choose whether to receive the output in the form of a text report or a formatted XML document.
- Control the output format by creating your own XSLT style sheets instead of using the ones provided by the **db2evmonfmt** command.

For example, the following command provides a unit of work report that:

1. Selects unit of work events that have occurred in the past 24 hours in the database SAMPLE. These event records are obtained from the unformatted event table called SAMPLE_UOW_EVENTS.
2. Provides formatted text output using the DB2EvmonUOW.xsl style sheet.

```
java db2evmonfmt -d SAMPLE -ue SAMPLE_UOW_EVENTS -ftext -ss DB2EvmonUOW.xsl -hours 24
```

Collecting unit of work event data and generating reports

You can use the unit of work event monitor to collect data about transactions that can be used for chargeback purposes. After the transaction event data has been collected in an unreadable form in an unformatted event table, this task describes how to obtain a readable text report.

Before you begin

To collect unit of work event monitor data, you must have SYSADM or SYSCTRL authority.

About this task

The unit of work event monitor collects relevant information that identifies application transactions and the corresponding CPU usage that can be used for chargeback purposes. For example, some of the information the unit of work event monitor collects for a transaction event is as follows:

- Total CPU usage time (TOTAL_CPU_TIME)
- Application handle (APPLICATION_HANDLE)

This task provides instructions for collecting unit of work event data for a given workload.

Restrictions

Input data values are not viewable if you do not have SYSADM or SYSCTRL authority.

Procedure

To collect detailed information regarding unit of work events, perform the following steps:

1. Create a unit of work event monitor called uowevmon by using the CREATE EVENT MONITOR FOR UNIT OF WORK statement, as shown in the following example:

   ```
   CREATE EVENT MONITOR uowevmon FOR UNIT OF WORK
      WRITE TO UNFORMATTED EVENT TABLE
   ```

2. Activate the unit of work event monitor called uowevmon by running the following statement:

   ```
   SET EVENT MONITOR uowevmon STATE 1
   ```

3. Enable the unit of work event data collection at the workload level using the ALTER WORKLOAD statement with statement history. To collect unit of work data for the FINANCE and PAYROLL applications, issue the following statements:

   ```
   ALTER WORKLOAD finance COLLECT UNIT OF WORK DATA BASE
   ALTER WORKLOAD payroll COLLECT UNIT OF WORK DATA BASE
   ```

4. Rerun the workload in order to collect unit of work transaction events.
5. Connect to the database.
6. Obtain the unit of work event report using the following approach:
   a. Use the XML parser tool, **db2evmonfmt**, to produce a flat-text report based on the event data collected in the unformatted event table, for example:

      ```
      java db2evmonfmt -d db_name -ue table_name -ftext -u user_id -p password
      ```

7. Analyze the report to determine how much CPU time applications are using so that appropriate charges can be billed.
8. If you want to turn OFF unit of work data collection for both FINANCE and PAYROLL applications, run the following statements:

   ```
   ALTER WORKLOAD finance COLLECT UNIT OF WORK DATA NONE
   ALTER WORKLOAD payroll COLLECT UNIT OF WORK DATA NONE
   ```

Example

The following is an example of a report obtained by using the **db2evmonfmt** Java-based report tool to convert data in the unformatted event table collected by the unit of work event monitor:

```
------------------------------------------------------
Event ID               : 1
Event Type             : UOW
Event Timestamp        : 2008-10-31-13.29.04.130849
Member of detection    : 0
------------------------------------------------------

Database Level Details
----------------------
Member Activation Time     : 2008-10-31T13:28:48.538973
Coordinator Member         : 0

Connection Level Details
------------------------
Application ID             : *LOCAL.gstager.081031172848
Application Handle         : 20
Application Name           : db2bp
Session Authorization ID   : GSTAGER
System Authorization ID    : GSTAGER
Connection Timestamp       : 2008-10-31T13:28:48.538973
Client Process ID          : 28167
```

```
Client Platform           : 30
Client Product ID         : SQL09070
Client Hostname           : gilera
Client Port Number        : 30143

UOW Level Details
-----------------------
Start Time                : 2008-10-31T13:28:51.560138
Stop Time                 : 2008-10-31T13:29:04.130849
Completion Status         : COMMIT
UOW ID                    : 5
Workoad Occurrence ID     : 1
Workload Name             : SYSDEFAULTUSERWORKLOAD
Workoad ID                : 1
Client userid             :
Client Workstation Name   :
Client Application Name   :
Client Accounting String  :
Local Transaction ID      : 00000000000000EB
Global Transaction ID     : 00000000000000000000000000000000000000000000
Log Space Used            : 0

UOW Metrics
-----------------------
TOTAL_CPU_TIME               : 7459
TOTAL_WAIT_TIME              : 0
ACT_ABORTED_TOTAL            : 0
...
```

Calculating the CPU time used by different applications or workloads with the unit of work event monitor:

This topic shows one way that you can use the unit of work event monitor in day-to-day database operations.

In some business environments, departments are billed for the processing time their applications use. You can use the unit of work event to record the CPU time used by different application, workloads, or service classes. This information can, in turn, be used in accounting applications that perform billing for system resources.

Before you begin

- You must have a table space in which to store event monitor output before you attempt to create the event monitor. A table space with a page size of at least 8k to store the unformatted event (UE) table produced by the event monitor is recommended. However, even with an 8k page size, the BLOB column used for storing the unformatted event data might not be stored inline within the table. If you want improved performance, consider using a table space with a large page size, such as 32k.
- For partitioned database environments, the table space must extend across all partitions.
- Unless a table space is explicitly named in the CREATE EVENT MONITOR command, the default table space for the database is used.

About this task

This task describes a basic scenario for “charge-back” accounting. In the example that follows, all work performed on the system is tracked. From the data gathered, reports are created that show the CPU time used by different applications.

Depending on how your organization is set up, tracking system time based on workload might be appropriate. Alternatively, you can also look at the CPU time used in different service super classes, by specific workloads, or even by different users. If the data is written to relational tables, as the example in this task shows, you can use SQL to query and present the data in almost limitless ways.

Note: Activities within a unit of work can run in different service subclasses. For this reason, it is not appropriate to aggregate unit of work information by service subclass. If you want to aggregate CPU time by service class, use the activity event monitor instead.

Procedure

1. Create a unit of work event monitor to capture information about units of work as they finish. For example, to create an event monitor called TRACKWORK, you might could use the following SQL:
   ```
   CREATE EVENT MONITOR TRACKWORK FOR UNIT OF WORK WRITE TO UNFORMATTED EVENT TABLE
   ```
   This statement creates a unit of work event monitor that writes to an unformatted event (UE) table. The UE table has the same name as the event monitor itself, TRACKWORK, and it is stored in the default table space.
2. Tell the database manager that you want to collect event information for all units of work completed on the database by running the following command:
   ```
   UPDATE DATABASE CONFIGURATION FOR dbname USING MON_UOW_DATA BASE
   ```
   This command causes information about all units of work executed on the data server to be sent to the active unit of work event monitors when the units of work complete. See “Configuring data collection” on page 95 for more information about controlling the scope of the unit of work data that is collected.
3. Next, activate the event monitor:
   ```
   SET EVENT MONITOR TRACKWORK STATE 1
   ```
   Note: By default, this event monitor starts automatically upon database activation, because the AUTOSTART option is applied by default. However, because this event monitor is being created in an already-active database, you must use the **SET EVENT MONITOR** command to start it manually.
   From this point on, the unit of work event monitor captures information for each unit of work as it runs to completion. As each unit of work completes, the event monitor adds a record for the event to the UE table TRACKWORK.
4. When you are ready to collect data for reporting purposes, you must extract the records from the TRACKWORK UE table.
   You can view this information in XML or relational format, using either the EVMON_FORMAT_UE_TO_XML or the EVMON_FORMAT_UE_TO_TABLES procedure to convert the data in the UE table. Alternatively, you can use the **db2evmonfmt** tool to create a text report of the information returned by the event monitor. This example shows the use of EVMON_FORMAT_UE_TO_TABLES to create relational tables that you can query in whatever way suits your needs.
   ```
   CALL EVMON_FORMAT_UE_TO_TABLES
      ('UOW', NULL, NULL, NULL, NULL, NULL, NULL, -1, 'SELECT * FROM TRACKWORK')
   ```
   The EVMON_FORMAT_UE_TO_TABLES procedure examines the UE table TRACKWORK produced by the event monitor; it selects each of the records from the UE table, and from them, creates rows containing the data collected by the unit of work event monitor in two relational tables:

- UOW_EVENT
- UOW_METRICS

The first table contains the most frequently used monitor elements and metrics associated with each event captured. The second contains detailed metrics for each event.

Notes:

- If you specify PKGLIST rather than BASE for the **MON_UOW_DATA** configuration parameter in step 2 on page 99, the EVMON_FORMAT_UE_TO_TABLES procedure creates a third table called UOW_PACKAGE_LIST. This table contains package list information related to the units of work. However, in this example, because only basic monitor elements are collected (see step 2 on page 99), this table will not contain any data. (See "Unit of work event monitor package listing information" on page 103 for more information about how the package list information can be used.)
- The values in the columns of UOW_METRICS can also be found in the XML document contained in the METRICS column of the UOW_EVENT table. They are provided in the UOW_METRICS table for more convenient, column-oriented access.

5. Query the tables produced in the previous step to see how CPU time was used by applications. The statement that follows returns a breakdown of total CPU time used by different users on the system since the unit of work event monitor was initialized. (This example assumes that client applications have identified themselves to the database using the sqleseti API, or through whatever application development environment you might be using, such as IBM® Rational® Application Developer for WebSphere® Software.

```
SELECT SUBSTR(E.CLIENT_USERID,1,10) AS CLIENT_ID,
       SUBSTR(E.CLIENT_APPLNAME,1,80) AS CLIENT_APP,
       SUBSTR(E.CLIENT_WRKSTNNAME,1,10) AS WKSTN,
       SUM(M.TOTAL_CPU_TIME) AS CPU_TIME
       FROM UOW_EVENT E, UOW_METRICS M
WHERE  M.APPLICATION_ID = E.APPLICATION_ID
   AND M.UOW_ID = E.UOW_ID
   AND M.MEMBER = E.MEMBER
GROUP BY E.CLIENT_USERID, E.CLIENT_APPLNAME, E.CLIENT_WRKSTNNAME
ORDER BY CPU_TIME DESC;
```

The preceding query returns the following results:

```
CLIENT_ID  CLIENT_APP                                                                      WKSTN      CPU_TIME
---------- ---------------------------------------------------------------------------- ---------- --------------------
                                                                                                          987770013
           DB2BATCH                                                                                       249375000
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003021324173                91181678
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1004201047173                66097348
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003191536588                28824420
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003191536434                27555568
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003221122075                16203116
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003221118191                15759227
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003221531062                15630121
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003221117466                15236718
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003221116141                14607249
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003251550366                14427883
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003051054311                 1312500
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003051053301                 1296875
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003051139066                 1296875
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003051152281                 1281250
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003041230283                 1046875
                                                                                           asrisk2          1031250
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003291503479                  515625
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003251506219                  484375
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003221444488                  453125
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003021323249                  406250
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003251544498                  296875
```

```
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003171431559                    171875
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003041227488                    156250
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003221117188                    109375
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003021333329                     62500
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003191502148                     62500
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003191527385                     62500
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003191528492                     62500
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003191530518                     62500
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003191533265                     62500
CLP C:\Documents and Settings\All Users\Application Data\IBM\DB2\DB2COPY1\DB2DAS                      62500
```

6. At this point, the unit of work event monitor TRACKWORK is still collecting information. Depending on how you want to track CPU time used by different applications, users or workloads, you can choose to take one of the following courses of action:

   - If you want to calculate CPU usage on a daily basis, you can leave this unit of work event monitor active. Each day, run the EVMON_FORMAT_UE_TO_TABLES procedure to retrieve just the time-spent metrics for the preceding day:

     ```
     CALL EVMON_FORMAT_UE_TO_TABLES
          ('UOW', NULL, NULL, NULL, NULL, NULL, NULL, -1,
           'SELECT * FROM TRACKWORK
           WHERE (DATE(EVENT_TIMESTAMP)=(CURRENT DATE - 1 DAY))'
          )
     ```

     With this approach, the three relational tables produced by the EVMON_FORMAT_UE_TO_TABLES procedure continue to grow, providing a history of CPU usage over time. The query in step 5 on page 100 returns the cumulative totals for CPU time since the tables were first created with the EVMON_FORMAT_UE_TO_TABLES procedure. You can modify that query to show only the results from the previous day as follows:

     ```
     SELECT SUBSTR(E.CLIENT_USERID,1,10) AS CLIENT_ID,
            SUBSTR(E.CLIENT_APPLNAME,1,80) AS CLIENT_APP,
            SUBSTR(E.CLIENT_WRKSTNNAME,1,10) AS WKSTN,
            SUM(M.TOTAL_CPU_TIME) AS CPU_TIME
            FROM UOW_EVENT E, UOW_METRICS M
     WHERE  M.APPLICATION_ID = E.APPLICATION_ID
        AND M.UOW_ID = E.UOW_ID
        AND M.MEMBER = E.MEMBER
        AND(DATE(E.EVENT_TIMESTAMP)=(CURRENT DATE - 1 DAY))
     GROUP BY E.CLIENT_USERID, E.CLIENT_APPLNAME, E.CLIENT_WRKSTNNAME
     ORDER BY CPU_TIME DESC;
     ```

     Tip: If you want to track CPU usage on a daily basis, but also want to manage how much data you collect on your system, remove data you no longer need from the UE table after you have updated the relational tables. For example, to delete the data collected on the previous day from the UE table TRACKWORK, use a DELETE statement similar to the one that follows:

     ```
     DELETE FROM TRACKWORK WHERE (DATE(EVENT_TIMESTAMP)=(CURRENT DATE - 1 DAY))
     ```

     While an event monitor is active, it holds an intention exclusive (IX) table lock on any tables to which it writes information to prevent those tables from being dropped while it is using them. When a large number of rows is being deleted, the DELETE statement acquires a large number of row locks. In this situation, lock escalation might occur, as row locks might be converted to a table lock. This request for table lock can cause the DELETE statement to hang, since the event monitor already has a lock on the table.

     To avoid this situation, consider setting a lock timeout before issuing the DELETE statement:

     ```
     SET CURRENT LOCK TIMEOUT 60
     ```

If increasing the lock timeout period does not resolve the problem, try deleting smaller subsets of the data, such as the records for smaller time periods (for example, 6 or 12 hours). This approach requires fewer locks, which will reduce the chance of lock escalation happening.

You can also prune the relational tables produced by EVMON_FORMAT_UE_TO_TABLES as needed to balance storage requirements with the need to view historical data.

- If you are finished calculating CPU time, you can stop the collection of event monitor information, and drop the event monitor and its related tables by performing the following steps:
  a. Disable the collection of unit of work for this event monitor information using the **SET EVENT MONITOR TRACKWORK STATE 0** command.
  b. Drop the event monitor itself using the DROP EVENT MONITOR statement.
  c. Drop the tables related to the event monitor using a DROP TABLE statement. In this case, there are four tables in total to drop:
     – TRACKWORK, the UE table used to collect information from the event monitor
     – UOW_EVENT
     – UOW_METRICS
     – UOW_PACKAGE_LIST
  d. Optional. If there are no remaining active event monitors, you might want to update the database configuration such that no unit of work event information is collected using the following command:

     ```
     UPDATE DATABASE CONFIGURATION FOR dbname USING MON_UOW_DATA NONE
     ```

Variation: Collecting metrics for specific workloads

The previous example illustrates how you can capture unit of work metrics for all work done on the system. Setting the scope of data collected using the **UPDATE DATABASE CONFIGURATION** command might cause more information to be collected than you need. You might, for example, want to track only work done by specific workloads. In this case, rather than enable collection of unit of work information across the whole database as shown in step 2 on page 99, you can specify the COLLECT UNIT OF WORK DATA clause with the CREATE or ALTER WORKLOAD statements. This clause causes only data for the workload specified to be collected by the event monitor. For example, to collect unit of work data for the workload named PAYROLL, use the following statement:

```
ALTER WORKLOAD PAYROLL COLLNECT UNIT OF WORK DATA BASE
```

You can collect data for multiple workloads by running an ALTER WORKLOAD statement for each.

The remaining steps are the same, except for step 5 on page 100, where you would change the query to resemble the one that follows:

```
SELECT E.WORKLOAD_NAME,
       SUM(M.TOTAL_CPU_TIME) AS CPU_TIME
       FROM UOW_EVENT E, UOW_METRICS M
WHERE  M.APPLICATION_ID = E.APPLICATION_ID
   AND M.UOW_ID = E.UOW_ID
   AND M.MEMBER = E.MEMBER
GROUP  BY E.WORKLOAD_NAME
ORDER  BY CPU_TIME DESC
```

The preceding statement reports the CPU time for each workload for which metrics collection is enabled:

```
WORKLOAD                       CPU_TIME
------------------------------ --------------------
PAYROLL                                  2143292042
MARKETING                                492784916

  2 record(s) selected.
```

Unit of work event monitor package listing information:

The unit of work event monitor can collect a listing of packages used within a unit of work. This information can be used to determine which stored procedures within an application might be taking more time than expected to run.

Starting with DB2 Version 9.7 Fix Pack 1, you can have information about the packages used within units of work included in the data the event monitor collects. This information is written to the unformatted event table along with the rest of the information associated with the event when the unit of work ends.

There are two ways to control the capture of this information:

- The PACKAGE LIST option for the COLLECT UNIT OF WORK DATA clause of the CREATE or ALTER WORKLOAD statements controls the collection of this information for *specific* workloads. If this option is specified, information for the units of work that are executed under the workload identified in the CREATE or ALTER WORKLOAD statements, including package list information, is sent to the active unit of work event monitors.
- The `PKGLIST` option for the **`mon_uow_data`** configuration parameter can be set so that information about *all* units of work executed on the data server, including package list information is sent to active unit of work event monitors.

The following data is collected for the package listing:

Package ID ("package_id - Package identifier monitor element" on page 790)
A unique ID that identifies a package.

Nesting level ("nesting_level - Nesting level monitor element" on page 768)
The level of nesting or recursion in effect when the statement was being run. Each level of nesting corresponds to nested or recursive invocation of a stored procedure or user-defined function (UDF).

Routine ID ("routine_id - Routine ID monitor element" on page 888)
A unique routine identifier. It returns zero if the activity is not part of a routine.

Invocation ID ("invocation_id - Invocation ID monitor element" on page 703)
An identifier that distinguishes one invocation of a routine from others at the same nesting level within a unit of work. It is unique within a unit of work for a specific nesting level.

Package elapsed time ("package_elapsed_time - Package elapsed time monitor element" on page 790)
The elapsed time spent executing sections within the package.

As the list of information collected for the package listing suggests, information is captured not only for each package but also for each invocation of a routine within a package.

Elapsed time is also tracked. The time calculated for a given invocation starts from the first execution of a section within a package until the database manager switches to another package. See "Examples" on page 106 to see more about how elapsed time is tracked.

How package lists are written to unformatted event tables

When you enable the collection of package list information, the unit of work event monitor writes two records to the unformatted event (UE) table for each unit of work. The first record contains the basic unit of work event monitor data. The next record contains the package listing information.

Package list information is stored in the UE table in a BLOB column. A list with 32 entries can be stored as an inline BLOB when the page size for the table space is 4k (the default). The number of entries that can be written to the package list is controlled by the **mon_pkglist_sz** configuration parameter. The default for this parameter is 32, which means that up to 32 entries can be included in the package listing. If you want to increase the number of entries that can be included in the package list, ensure that the UE table used to store the event monitor output is created in a table space with a larger page size. Assume that every increase of 32 in the size of the package list requires an increase of 4k in the page size of the table space. So, for example, if you want to have up to 64 entries in the package list, ensure that the page size for the table space is at least 8k. If you increase **mon_pkglist_sz** without increasing the page size of the table space, the package list is still created, however the BLOB is not stored inline in the table, which might affect performance.

Note: You can use the ADMIN_IS_INLINED administrative function to determine whether the BLOB that contains the package list information is stored inline.

Package listing output

As stated earlier, when collecting package information, the unit of work event monitor writes two records to the UE table. Each of the interfaces for displaying the data in a UE table provides a mechanism for viewing the information contained in the two records together. For example, the **db2evmonfmt** tool combines the information in each record into a single report. If you use the EVMON_FORMAT_UE_TO_TABLES procedure, it produces two relational tables that you can join; EVMON_FORMAT_UE_TO_XML produces a single XML document that contains the information from both records. For more information, see "Accessing event data captured by a unit of work event monitor" on page 95.

Note: In a partitioned database environment, the package list is only reported in the unit of work event generated by the *coordinator agent* and reflects the time spent in each package by that agent specifically; it does not reflect time spent in those packages by any other agent at any other partition.

Figure 4 on page 105 shows the information produced by the unit of work event monitor, as formatted by the **db2evmonfmt** tool.

```
------------------------------------------------------
Event ID               : 12
Event Type             : UOW
Event Timestamp        : 2009-12-08-14.44.39.162707
Member                 : 0
Release                : 9070200
------------------------------------------------------

Database Level Details
----------------------
Database Member Activation Time : 2009-12-08-14.41.55.089416
Coordinator Member              : 0

Connection Level Details
------------------------
Application ID                  : *LOCAL.gstager.091208194155
Application Handle              : 21
Application Name                : db2bp
Session Authorization ID        :
System Authorization ID         :
Connection Timestamp            : 2009-12-08-14.41.55.089416
Client Process ID               : 13043
Client Platform                 : LINUXX8664
Client Product ID               : SQL09072
Client Protocol                 : LOCAL
Client Hostname                 : HOSTX
Client Port Number              : 0

UOW Level Details
-------------------------
Start Time                      : 2009-12-08-14.44.39.160651
Stop Time                       : 2009-12-08-14.44.39.162707
Completion Status               : COMMIT
UOW ID                          : 12
Workoad Occurrence ID           : 1
Workload Name                   : SYSDEFAULTUSERWORKLOAD
Workoad ID                      : 1
Service Superclass Name         : SYSDEFAULTUSERCLASS
Service Subclass Name           : SYSDEFAULTSUBCLASS
Service Class ID                : 13
Client Userid                   :
Client Workstation Name         :
Client Application Name         :
Client Accounting String        :
Local Transaction ID            : 000000000000013B
Global Transaction ID           : 00000000000000000000000000000000000000000000
Log Space Used                  : 124

UOW Metrics
-------------------------
TOTAL_CPU_TIME            : 1591
TOTAL_WAIT_TIME           : 8363
ACT_ABORTED_TOTAL         : 0
ACT_COMPLETED_TOTAL       : 1
ACT_REJECTED_TOTAL:       : 0
AGENT_WAIT_TIME           : 87
AGENT_WAITS_TOTAL         : 1
APP_RQSTS_COMPLETED_TOTAL : 1
    .
    .
    .

Package List
-------------------------
Package List Size       : 2
Package List Exceeded   : no

PACKAGE_ID           NESTING_LEVEL ROUTINE_ID  INVOCATION_ID        PACKAGE_ELAPSED_TIME
-------------------- ------------- ----------- -------------------- --------------------
240                  0             0           0                    0
330                  1             66539       1                    1
```

Note: Some of the metrics in the "UOW Metrics" section have been excluded.

Figure 4. Sample output from the unit of work event monitor, with package listing information

The number of packages that appear in the package list for a given unit of work is reflected in the **package_list_count** monitor element ("Package List Size" in the preceding report), which is included with the base unit of work event monitor data. If the number of packages used with the unit of work exceeds the value

specified in the **mon_pkglist_sz** configuration parameter, the additional packages are not included in the package listing. However, the **package_list_exceeded** monitor element indicates whether there were more packages than would fit into the package list. This monitor element is returned along with the base information for the unit of work event monitor ("Package List Exceeded" in Figure 4 on page 105). If the value for this monitor element is YES, you can increase the value for **mon_pkglist_sz** to have a larger number of packages included in the package list.

Examples

Each of the examples that follow show the information returned for the package listing as it would be displayed by the **db2evmonfmt** tool.

Example 1: An application that executes one or more sections in a single package

In this example, one package with a package ID of 300 was run for this unit of work.

```
PACKAGE_ID  NESTING_LEVEL  ROUTINE_ID INVOCATION_ID ELAPSED_TIME
----------  -------------  ---------- ------------- ------------
300         0              0          0             100
```

In this case, there is one entry on the package list, which reflects the execution of one or more sections in the package. All sections executed from the same package are considered to be part of the same package invocation.

Example 2: An application calls a stored procedure in a package

In this example, the package with a package ID of 300 calls a stored procedure with an ID of 806. Three sections are executed within the stored procedure.

```
PACKAGE_ID  NESTING_LEVEL  ROUTINE_ID INOVATION_ID ELAPSED_TIME
----------  -------------  ---------- ------------ ------------
300         0              0          0            21
300         1              806        1            100
```

This output shows two entries in the list. One entry is for the call to the stored procedure, and one for the execution of the three sections within the stored procedure. The NESTING_LEVEL for the second entry in the list reflects the fact that the stored procedure was called from another package.

Example 3: An application executes sections in two different packages

In this example, an application executes sections from one package, then another package, and then back to the first package. No stored procedures are called. The pseudocode that follows is a representation of this unit of work:

```
Application
    EXEC PACKAGEA
    EXEC PACKAGEB
    EXEC PACKAGEA
```

Assume also that each of these invocations require 100 ms, 25 ms, and 460 ms, respectively. The following output shows what the package listing would look like:

```
PACKAGE_ID  NESTING_LEVEL  ROUTINE_ID INVOCATION_ID ELAPSED_TIME
----------  -------------  ---------- ------------- ------------
300         0              0          0             560
301         0              0          0             25
```

In this case, there are two entries in the list. Package A, with PACKAGE_ID 300 had sections that ran for 560 ms in total. Package B ran

for 25 ms. Package A is represented by a single line because each invocation has the same INVOCATION_ID and NESTING_LEVEL. INVOCATION_ID and NESTING_LEVEL remain at 0, because no stored procedures were called in either package.

Example 4: An application executes sections and stored procedures in multiple packages

In this example, there are 3 packages with IDs 100, 101, and 102. The application is in package 100. There are two stored procedures with IDs 201 and 202. The first stored procedure (SP1) is in package 101, and the second (SP2) is in package 102. The pseudocode that follows is a representation of this unit of work:

```
Application
   CALL SP1 a
      INSERT INTO T1 VALUES(7)  b
      CALL SP2 c
         INSERT INTO T2 VALUES(8)
      CALL SP2 d
         INSERT INTO T2 VALUES(8)
```

The package listing for this unit of work would be as follows:

```
PACKAGE_ID  NESTING_LEVEL  ROUTINE_ID INVOCATION_ID ELAPSED_TIME
----------  -------------  ---------- ------------- ------------
100         0              0          0             21
101         1 1            201        1             40
102         2 2            202        1 3           35
102         2              202        2 3           35
```

In the preceding output, there are four entries:

- The first corresponds to the execution of the call to SP1 within the first package, line **a** in the pseudocode that represents the unit of work.
- The second corresponds to the execution of the sections within the stored procedure with ID 201 in package 101. These sections include lines **b**, **c**, and **d**. The nesting level increases to 1, as shown by **1**.
- The third entry represents the execution of the first of the INSERT INTO T2 statements in SP2, as called from SP1. The nesting level increases again (**2**).
- The fourth entry in the list represents the execution of the second of the INSERT INTO T2 statements in SP2. The nesting level remains the same, because like the previous call to SP2, this stored procedure is called from SP1. However, because these two statements occur within separate invocations of the stored procedure, they each have separate invocation IDs (**3**). Thus, there are two separate entries in the package listing.

Information written to XML for a unit of work event monitor

Information written for a unit of work event monitor from the EVMON_FORMAT_UE_TO_XML table function. This is also documented in the `sqllib/misc/DB2EvmonUOW.xsd` file.

db2_uow_event

The main schema that describes a unit of work event.

Element content: (completion_status , start_time , stop_time , connection_time , application_name , application_handle , application_id , uow_id , workload_occurrence_id , coord_member , member_activation_time , workload_name , workload_id , service_superclass_name {zero or one times (?)} , service_subclass_name {zero or one times (?)} , service_class_id {zero or one times

(?)} , session_authid {zero or one times (?)} , system_authid , client_pid , client_product_id , client_platform , client_protocol {zero or one times (?)} , client_userid {zero or one times (?)} , client_wrkstnname {zero or one times (?)} , client_applname {zero or one times (?)} , client_acctng {zero or one times (?)} , local_transaction_id , global_transaction_id , system_metrics , client_hostname , client_port_number , uow_log_space_used , package_list , ANY content (skip) {zero or more (*)})

Attributes:

QName	Type	Fixed	Default	Use	Annotation
id	xs:long			required	
type	xs:string - Max length: 32 (UOW)			required	
timestamp	xs:dateTime			required	
member	member_type			required	
release	xs:long			required	
ANY attribute from ANY namespace					

package_id

>

See monitor element package_id for more details.

Contained by: package_entry

Element content:

Type	Facet
xs:long	

package_elapsed_time

See monitor element package_elapsed_time for more details.

Contained by: package_entry

Element content:

Type	Facet
xs:long	

invocation_id

See monitor element invocation_id for more details.

Contained by: package_entry

Element content:

Type	Facet
xs:int	

routine_id

See monitor element routine_id for more details.

Contained by: package_entry

Element content:

Type	Facet
xs:int	

nesting_level

See monitor element nesting_level for more details.

Contained by: package_entry

Element content:

Type	Facet
xs:int	

package_entry

Contained by: package_list_entries

Element content: (package_id , package_elapsed_time , invocation_id , routine_id , nesting_level , ANY content (skip) {zero or more (*)})

Attributes:

QName	Type	Fixed	Default	Use	Annotation
ANY attribute from ANY namespace					

package_list_size

Contained by: package_list

Element content:

Type	Facet
xs:int	

package_list_exceeded

See monitor element package_list_exceeded for more information.

Contained by: package_list

Element content:

Type	Facet
xs:string	Max length: 3

package_list_entries

Contained by: package_list

Element content: (package_entry {zero or more (*)})

Attributes:

QName	Type	Fixed	Default	Use	Annotation
ANY attribute from ANY namespace					

completion_status

The completion status of the unit of work. Possible values are: UNKNOWN, COMMIT, ROLLBACK, GLOBAL_COMMIT, GLOBAL ROLLBACK, XA_END, XA_PREPARE

Contained by: db2_uow_event

Element content:

Type	Facet
xs:string	Max length: 128

start_time

The start time of the unit of work. See monitor element uow_start_time for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:dateTime	

stop_time

The stop time of the unit of work. See monitor element uow_stop_time for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:dateTime	

connection_time

The time the application connected to the database member. See monitor element conn_time for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:dateTime	

application_name

The name of the application running at the client, as known to the database. See monitor element appl_name for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:string	Max length: 128

application_handle

A system-wide unique ID for the application. See monitor element agent_id for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:long	

application_id

This identifier is generated when the application connects to the database at the database manager. See monitor element appl_id for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:string	Max length: 128

uow_id

The unit of work ID to which this activity record applies. See monitor element uow_id for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:int	

workload_occurrence_id

The workload occurrence ID to which this activity record applies. See monitor element workload_occurrence_id for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:int	

coord_member

See monitor element coord_member for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:short	Max inclusive: 999

member_activation_time

The time this database member was activated. See monitor element db_conn_time for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:dateTime	

workload_name

The name of the workload under which the unit of work completed. See monitor element workload_name for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:string	Max length: 128

workload_id

The workload ID of the workload under which the unit of work completed. See monitor element workload_id for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:int	

service_superclass_name

The name of the service super class under which the unit of work completed. See monitor element service_superclass_name for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:string	Max length: 128

service_subclass_name

The name of the service sub class under which the unit of work completed. See monitor element service_subclass_name for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:string	Max length: 128

service_class_id

The service class ID of the service class under which the unit of work completed. See monitor element service_class_id for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:int	

session_authid

The session authorization ID of the user who invoked the application that is being monitored. See monitor element session_auth_id for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:string	Max length: 128

system_authid

The system authorization ID of the user who invoked the application that is being monitored. See monitor element system_auth_id for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:string	Max length: 128

client_pid

The process ID reported by the client. See monitor element client_pid for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:long	

client_product_id

The product ID of the client. See monitor element client_prdid for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:string	Max length: 128

client_platform

The platform of the client. See monitor element client_platform for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:string	Max length: 12

Attributes:

QName	Type	Fixed	Default	Use	Annotation
id	xs:short			optional	

client_protocol

The product ID of the client. See monitor element client_protocol for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:string	Max length: 10

client_userid

The client user ID generated by a transaction manager and provided to the server. See monitor element client_userid for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:string	Max length: 255

client_wrkstnname

Identifies the client system or workstation, if the sqleseti API was issued in this connection. See monitor element client_wrkstnname for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:string	Max length: 255

client_applname

Identifies the server transaction program performing the transaction, if the sqleseti API was issued in this connection. See monitor element client_applname for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:string	Max length: 255

client_acctng

The data passed to the target database for logging and diagnostic purposes, if the sqleseti API was issued in this connection. See monitor element client_acctng for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:string	Max length: 255

local_transaction_id

The local transaction id for the unit of work.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:hexBinary	Max length: 8

global_transaction_id

The global transaction ID for the unit of work.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:hexBinary	Max length: 20

system_metrics

The metrics for the unit of work.

Contained by: db2_uow_event

Element content: (wlm_queue_time_total , wlm_queue_assignments_total , fcm_tq_recv_wait_time , fcm_message_recv_wait_time , fcm_tq_send_wait_time , fcm_message_send_wait_time , agent_wait_time , agent_waits_total , lock_wait_time , lock_waits , direct_read_time , direct_read_reqs , direct_write_time , direct_write_reqs , log_buffer_wait_time , num_log_buffer_full , log_disk_wait_time , log_disk_waits_total , tcpip_recv_wait_time , tcpip_recvs_total , client_idle_wait_time , ipc_recv_wait_time , ipc_recvs_total , ipc_send_wait_time , ipc_sends_total , tcpip_send_wait_time , tcpip_sends_total , pool_write_time , pool_read_time , audit_file_write_wait_time , audit_file_writes_total , audit_subsystem_wait_time , audit_subsystem_waits_total , diaglog_write_wait_time , diaglog_writes_total , fcm_send_wait_time , fcm_recv_wait_time , total_wait_time , total_rqst_time , rqsts_completed_total , total_app_rqst_time , app_rqsts_completed_total , total_section_sort_proc_time , total_section_sort_time , total_section_sorts , rows_read , rows_modified , pool_data_l_reads , pool_index_l_reads , pool_temp_data_l_reads , pool_temp_index_l_reads , pool_xda_l_reads , pool_temp_xda_l_reads , total_cpu_time , act_completed_total , pool_data_p_reads , pool_temp_data_p_reads , pool_xda_p_reads , pool_temp_xda_p_reads , pool_index_p_reads , pool_temp_index_p_reads , pool_data_writes , pool_xda_writes , pool_index_writes , direct_reads , direct_writes , rows_returned , deadlocks , lock_timeouts , lock_escals , fcm_sends_total , fcm_recvs_total , fcm_send_volume , fcm_recv_volume , fcm_message_sends_total , fcm_message_recvs_total , fcm_message_send_volume , fcm_message_recv_volume , fcm_tq_sends_total , fcm_tq_recvs_total , fcm_tq_send_volume , fcm_tq_recv_volume , tq_tot_send_spills , tcpip_send_volume , tcpip_recv_volume , ipc_send_volume , ipc_recv_volume , post_threshold_sorts , post_shrthreshold_sorts , sort_overflows , audit_events_total , total_rqst_mapped_in {zero or one times (?)} , total_rqst_mapped_out {zero or one times (?)} , act_rejected_total , act_aborted_total , total_sorts , total_routine_time , total_compile_proc_time , total_compile_time , total_compilations , total_implicit_compile_proc_time , total_implicit_compile_time , total_implicit_compilations , total_runstats_proc_time , total_runstats_time , total_runstats , total_reorg_proc_time , total_reorg_time , total_reorgs , total_load_proc_time , total_load_time , total_loads , total_section_proc_time , total_section_time , total_app_section_executions , total_commit_proc_time , total_commit_time , total_app_commits , total_rollback_proc_time , total_rollback_time , total_app_rollbacks , total_routine_user_code_proc_time , total_routine_user_code_time , thresh_violations , num_lw_thresh_exceeded ,

total_routine_invocations , int_commits , int_rollbacks , cat_cache_inserts , cat_cache_lookups , pkg_cache_inserts , pkg_cache_lookups , act_rqsts_total , total_act_wait_time , total_act_time , ANY content (skip) {zero or more (*)})

Attributes:

QName	Type	Fixed	Default	Use	Annotation
release	xs:long			required	
ANY attribute from ANY namespace					

client_hostname

The host name of the client. See monitor element client_hostname for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:string	Max length: 255

client_port_number

The port number of the client. See monitor element client_port_number for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:int	

uow_log_space_used

The amount of log space used during the unit of work. See monitor element uow_log_space_used for more details.

Contained by: db2_uow_event

Element content:

Type	Facet
xs:long	

package_list

The package list for the unit of work.

Contained by: db2_uow_event

Element content: (package_list_size , package_list_exceeded , package_list_entries , ANY content (skip) {zero or more (*)})

Attributes:

QName	Type	Fixed	Default	Use	Annotation
ANY attribute from ANY namespace					

Information written to relational tables for a unit of work event monitor

Information written for a unit of work event monitor from the EVMON_FORMAT_UE_TO_TABLES table function. This is also documented in the `sqllib/misc/DB2EvmonUOW.xsd` file.

Table 11. Information returned for a unit of work event monitor: Table name: UOW_EVENT

Column Name	Data Type	Description
EVENT_ID	INTEGER NOT NULL	
TYPE	VARCHAR(128) NOT NULL	
EVENT_TIMESTAMP	TIMESTAMP NOT NULL	
MEMBER	SMALLINT	"member - Database member monitor element" on page 760
COORD_MEMBER	SMALLINT	"coord_member - Coordinator member monitor element" on page 590
COMPLETION_STATUS	VARCHAR(128)	"completion_status - Completion status monitor element" on page 566
START_TIME	TIMESTAMP	"start_time - Event Start Time" on page 927
STOP_TIME	TIMESTAMP	"stop_time - Event Stop Time" on page 947
WORKLOAD_NAME	VARCHAR(128)	"workload_name - Workload name monitor element" on page 1064
WORKLOAD_ID	INTEGER	"workload_id - Workload ID monitor element" on page 1063
SERVICE_SUPERCLASS_NAME	VARCHAR(128)	"service_superclass_name - Service superclass name monitor element" on page 909
SERVICE_SUBCLASS_NAME	VARCHAR(128)	"service_subclass_name - Service subclass name monitor element" on page 908
SERVICE_CLASS_ID	INTEGER	"service_class_id - Service class ID monitor element" on page 907
UOW_ID	INTEGER	"uow_id - Unit of work ID monitor element" on page 1048
WORKLOAD_OCCURRENCE_ID	INTEGER	"workload_occurrence_id - Workload occurrence identifier monitor element" on page 1065
CONNECTION_TIME	TIMESTAMP	
MEMBER_ACTIVATION_TIME	TIMESTAMP	

Table 11. Information returned for a unit of work event monitor: Table name: UOW_EVENT (continued)

Column Name	Data Type	Description
APPLICATION_ID	VARCHAR(128)	
APPLICATION_HANDLE	BIGINT	"application_handle - Application handle monitor element" on page 528
APPLICATION_NAME	VARCHAR(128)	
SYSTEM_AUTHID	VARCHAR(128)	
SESSION_AUTHID	VARCHAR(128)	
CLIENT_PLATFORM	VARCHAR(12)	"client_platform - Client operating platform monitor element" on page 559
CLIENT_PID	BIGINT	"client_pid - Client process ID monitor element" on page 558
CLIENT_PRODUCT_ID	VARCHAR(128)	
CLIENT_PROTOCOL	VARCHAR(10)	"client_protocol - Client communication protocol monitor element" on page 561
CLIENT_HOSTNAME	VARCHAR(255)	"client_hostname - Client hostname monitor element" on page 557
CLIENT_PORT_NUMBER	INTEGER	"client_port_number - Client port number monitor element" on page 560
CLIENT_WRKSTNNAME	VARCHAR(255)	"client_wrkstnname - Client workstation name monitor element" on page 563
CLIENT_ACCTNG	VARCHAR(255)	"client_acctng - Client accounting string monitor element" on page 554
CLIENT_USERID	VARCHAR(255)	"client_userid - Client user ID monitor element" on page 562
CLIENT_APPLNAME	VARCHAR(255)	"client_applname - Client application name monitor element" on page 555
LOCAL_TRANSACTION_ID	VARCHAR(16)	
GLOBAL_TRANSACTION_ID	VARCHAR(40)	
UOW_LOG_SPACE_USED	BIGINT	"uow_log_space_used - Unit of work log space used monitor element" on page 1049
PACKAGE_LIST_SIZE	INTEGER	
PACKAGE_LIST_EXCEEDED	CHAR(3)	
METRICS	BLOB(1M)	XML document containing metrics-related monitor elements. The metrics in this document are the same as those described in the UOW_METRICS table that appears later in this topic. See Chapter 3, "Interfaces that return monitor data in XML documents," on page 9 for more information.

Table 12. Information returned for a unit of work event monitor: Table name: UOW_PACKAGE_LIST

Column Name	Data Type	Description
MEMBER	SMALLINT	"member - Database member monitor element" on page 760

Table 12. Information returned for a unit of work event monitor: Table name: UOW_PACKAGE_LIST (continued)

Column Name	Data Type	Description
UOW_ID	INTEGER	"uow_id - Unit of work ID monitor element" on page 1048
APPLICATION_ID	VARCHAR(128)	
PACKAGE_ID	BIGINT	
NESTING_LEVEL	INTEGER	"nesting_level - Nesting level monitor element" on page 768
ROUTINE_ID	INTEGER	"routine_id - Routine ID monitor element" on page 888
INVOCATION_ID	INTEGER	"invocation_id - Invocation ID monitor element" on page 703
PACKAGE_ELAPSED_TIME	BIGINT	

Table 13. Information returned for a unit of work event monitor: Table name: UOW_METRICS. The metrics in this table are the same as those returned in the METRICS monitor element in the UOW_EVENT table

Column Name	Data Type	Description
MEMBER	SMALLINT	"member - Database member monitor element" on page 760
UOW_ID	INTEGER	"uow_id - Unit of work ID monitor element" on page 1048
APPLICATION_ID	VARCHAR(128)	
ACT_ABORTED_TOTAL	BIGINT	"act_aborted_total - Total aborted activities monitor element" on page 494
ACT_COMPLETED_TOTAL	BIGINT	"act_completed_total - Total completed activities monitor element" on page 495
ACT_REJECTED_TOTAL	BIGINT	"act_rejected_total - Total rejected activities monitor element" on page 497
AGENT_WAIT_TIME	BIGINT	"agent_wait_time - Agent wait time monitor element" on page 511
AGENT_WAITS_TOTAL	BIGINT	"agent_waits_total - Total agent waits monitor element" on page 512
POOL_DATA_L_READS	BIGINT	"pool_data_l_reads - Buffer pool data logical reads monitor element" on page 816
POOL_INDEX_L_READS	BIGINT	"pool_index_l_reads - Buffer pool index logical reads monitor element" on page 825
POOL_TEMP_DATA_L_READS	BIGINT	"pool_temp_data_l_reads - Buffer pool temporary data logical reads monitor element" on page 835
POOL_TEMP_INDEX_L_READS	BIGINT	"pool_temp_index_l_reads - Buffer pool temporary index logical reads monitor element" on page 839
POOL_TEMP_XDA_L_READS	BIGINT	"pool_temp_xda_l_reads - Buffer pool temporary XDA data logical reads monitor element" on page 843

Table 13. Information returned for a unit of work event monitor: Table name: UOW_METRICS (continued). The metrics in this table are the same as those returned in the METRICS monitor element in the UOW_EVENT table

Column Name	Data Type	Description
POOL_XDA_L_READS	BIGINT	"pool_xda_l_reads - Buffer pool XDA data logical reads monitor element" on page 849
POOL_DATA_P_READS	BIGINT	"pool_data_p_reads - Buffer pool data physical reads monitor element" on page 818
POOL_INDEX_P_READS	BIGINT	"pool_index_p_reads - Buffer pool index physical reads monitor element" on page 827
POOL_TEMP_DATA_P_READS	BIGINT	"pool_temp_data_p_reads - Buffer pool temporary data physical reads monitor element" on page 837
POOL_TEMP_INDEX_P_READS	BIGINT	"pool_temp_index_p_reads - Buffer pool temporary index physical reads monitor element" on page 841
POOL_TEMP_XDA_P_READS	BIGINT	"pool_temp_xda_p_reads - Buffer pool temporary XDA data physical reads monitor element" on page 845
POOL_XDA_P_READS	BIGINT	"pool_xda_p_reads - Buffer pool XDA data physical reads monitor element" on page 851
POOL_DATA_WRITES	BIGINT	"pool_data_writes - Buffer pool data writes monitor element" on page 820
POOL_INDEX_WRITES	BIGINT	"pool_index_writes - Buffer pool index writes monitor element" on page 829
POOL_XDA_WRITES	BIGINT	"pool_xda_writes - Buffer pool XDA data writes monitor element" on page 853
POOL_READ_TIME	BIGINT	"pool_read_time - Total buffer pool physical read time monitor element" on page 833
POOL_WRITE_TIME	BIGINT	"pool_write_time - Total buffer pool physical write time monitor element" on page 847
CLIENT_IDLE_WAIT_TIME	BIGINT	"client_idle_wait_time - Client idle wait time monitor element" on page 557
DEADLOCKS	BIGINT	"deadlocks - Deadlocks detected monitor element" on page 614
DIRECT_READS	BIGINT	"direct_reads - Direct reads from database monitor element" on page 624
DIRECT_READ_TIME	BIGINT	"direct_read_time - Direct read time monitor element" on page 622
DIRECT_WRITES	BIGINT	"direct_writes - Direct writes to database monitor element" on page 629
DIRECT_WRITE_TIME	BIGINT	"direct_write_time - Direct write time monitor element" on page 627
DIRECT_READ_REQS	BIGINT	"direct_read_reqs - Direct read requests monitor element" on page 620

Table 13. Information returned for a unit of work event monitor: Table name: UOW_METRICS (continued). The metrics in this table are the same as those returned in the METRICS monitor element in the UOW_EVENT table

Column Name	Data Type	Description
DIRECT_WRITE_REQS	BIGINT	"direct_write_reqs - Direct write requests monitor element" on page 626
FCM_RECV_VOLUME	BIGINT	"fcm_recv_volume - FCM received volume monitor element" on page 647
FCM_RECVS_TOTAL	BIGINT	"fcm_recvs_total - FCM receives total monitor element" on page 650
FCM_SEND_VOLUME	BIGINT	"fcm_send_volume - FCM send volume monitor element" on page 651
FCM_SENDS_TOTAL	BIGINT	"fcm_sends_total - FCM sends total monitor element" on page 654
FCM_RECV_WAIT_TIME	BIGINT	"fcm_recv_wait_time - FCM received wait time monitor element" on page 648
FCM_SEND_WAIT_TIME	BIGINT	"fcm_send_wait_time - FCM send wait time monitor element" on page 652
IPC_RECV_VOLUME	BIGINT	"ipc_recv_volume - Interprocess communication received volume monitor element" on page 704
IPC_RECV_WAIT_TIME	BIGINT	"ipc_recv_wait_time - Interprocess communication received wait time monitor element" on page 705
IPC_RECVS_TOTAL	BIGINT	"ipc_recvs_total - Interprocess communication receives total monitor element" on page 706
IPC_SEND_VOLUME	BIGINT	"ipc_send_volume - Interprocess communication send volume monitor element" on page 707
IPC_SEND_WAIT_TIME	BIGINT	"ipc_send_wait_time - Interprocess communication send wait time monitor element" on page 708
IPC_SENDS_TOTAL	BIGINT	"ipc_sends_total - Interprocess communication send total monitor element" on page 709
LOCK_ESCALS	BIGINT	"lock_escals - Number of lock escalations monitor element" on page 721
LOCK_TIMEOUTS	BIGINT	"lock_timeouts - Number of lock timeouts monitor element" on page 732
LOCK_WAIT_TIME	BIGINT	"lock_wait_time - Time waited on locks monitor element" on page 734
LOCK_WAITS	BIGINT	"lock_waits - Lock waits monitor element" on page 737
LOG_BUFFER_WAIT_TIME	BIGINT	"log_buffer_wait_time - Log buffer wait time monitor element" on page 740
NUM_LOG_BUFFER_FULL	BIGINT	"num_log_buffer_full - Number of times full log buffer caused agents to wait monitor element" on page 775
LOG_DISK_WAIT_TIME	BIGINT	"log_disk_wait_time - Log disk wait time monitor element" on page 741

Table 13. Information returned for a unit of work event monitor: Table name: UOW_METRICS (continued). The metrics in this table are the same as those returned in the METRICS monitor element in the UOW_EVENT table

Column Name	Data Type	Description
LOG_DISK_WAITS_TOTAL	BIGINT	"log_disk_waits_total - Total log disk waits monitor element" on page 743
RQSTS_COMPLETED_TOTAL	BIGINT	"rqsts_completed_total - Total requests completed monitor element" on page 897
ROWS_MODIFIED	BIGINT	"rows_modified - Rows modified monitor element" on page 890
ROWS_READ	BIGINT	"rows_read - Rows read monitor element" on page 892
ROWS_RETURNED	BIGINT	"rows_returned - Rows returned monitor element" on page 893
TCPIP_RECV_VOLUME	BIGINT	"tcpip_recv_volume - TCP/IP received volume monitor element" on page 975
TCPIP_SEND_VOLUME	BIGINT	"tcpip_send_volume - TCP/IP send volume monitor element" on page 978
TCPIP_RECV_WAIT_TIME	BIGINT	"tcpip_recv_wait_time - TCP/IP received wait time monitor element" on page 976
TCPIP_RECVS_TOTAL	BIGINT	"tcpip_recvs_total - TCP/IP receives total monitor element" on page 977
TCPIP_SEND_WAIT_TIME	BIGINT	"tcpip_send_wait_time - TCP/IP send wait time monitor element" on page 979
TCPIP_SENDS_TOTAL	BIGINT	"tcpip_sends_total - TCP/IP sends total monitor element" on page 980
TOTAL_APP_RQST_TIME	BIGINT	"total_app_rqst_time - Total application request time monitor element" on page 992
TOTAL_RQST_TIME	BIGINT	"total_rqst_time - Total request time monitor element" on page 1023
WLM_QUEUE_TIME_TOTAL	BIGINT	"wlm_queue_time_total - Workload manager total queue time monitor element" on page 1060
WLM_QUEUE_ASSIGNMENTS_TOTAL	BIGINT	"wlm_queue_assignments_total - Workload manager total queue assignments monitor element" on page 1059
TOTAL_CPU_TIME	BIGINT	"total_cpu_time - Total CPU time monitor element" on page 1001
TOTAL_WAIT_TIME	BIGINT	"total_wait_time - Total wait time monitor element" on page 1038
APP_RQSTS_COMPLETED_TOTAL	BIGINT	"app_rqsts_completed_total - Total application requests completed monitor element" on page 518
TOTAL_SECTION_SORT_TIME	BIGINT	"total_section_sort_time - Total section sort time monitor element" on page 1030
TOTAL_SECTION_SORT_PROC_TIME	BIGINT	"total_section_sort_proc_time - Total section sort processing time monitor element" on page 1028

Table 13. Information returned for a unit of work event monitor: Table name: UOW_METRICS (continued). The metrics in this table are the same as those returned in the METRICS monitor element in the UOW_EVENT table

Column Name	Data Type	Description
TOTAL_SECTION_SORTS	BIGINT	"total_section_sorts - Total section sorts monitor element" on page 1032
TOTAL_SORTS	BIGINT	"total_sorts - Total sorts monitor element" on page 1036
POST_THRESHOLD_SORTS	BIGINT	"post_threshold_sorts - Post threshold sorts monitor element" on page 858
POST_SHRTHRESHOLD_SORTS	BIGINT	"post_shrthreshold_sorts - Post shared threshold sorts monitor element" on page 856
SORT_OVERFLOWS	BIGINT	"sort_overflows - Sort overflows monitor element" on page 915
TOTAL_COMPILE_TIME	BIGINT	"total_compile_time - Total compile time monitor element" on page 999
TOTAL_COMPILE_PROC_TIME	BIGINT	"total_compile_proc_time - Total compile processing time monitor element" on page 998
TOTAL_COMPILATIONS	BIGINT	"total_compilations - Total compilations monitor element" on page 997
TOTAL_IMPLICIT_COMPILE_TIME	BIGINT	"total_implicit_compile_time - Total implicit compile time monitor element" on page 1005
TOTAL_IMPLICIT_COMPILE_PROC_ TIME	BIGINT	"total_implicit_compile_proc_time - Total implicit compile processing time monitor element" on page 1004
TOTAL_IMPLICIT_COMPILATIONS	BIGINT	"total_implicit_compilations - Total implicit complications monitor element" on page 1003
TOTAL_SECTION_TIME	BIGINT	"total_section_time - Total section time monitor element" on page 1033
TOTAL_SECTION_PROC_TIME	BIGINT	"total_section_proc_time - Total section processing time monitor element" on page 1027
TOTAL_APP_SECTION_EXECUTIONS	BIGINT	"total_app_section_executions - Total application section executions monitor element" on page 993
TOTAL_ACT_TIME	BIGINT	"total_act_time - Total activity time monitor element" on page 988
TOTAL_ACT_WAIT_TIME	BIGINT	"total_act_wait_time - Total activity wait time monitor element" on page 989
ACT_RQSTS_TOTAL	BIGINT	"act_rqsts_total - Total activity requests monitor elements" on page 500
TOTAL_ROUTINE_TIME	BIGINT	"total_routine_time - Total routine time monitor element" on page 1018
TOTAL_ROUTINE_INVOCATIONS	BIGINT	"total_routine_invocations - Total routine invocations monitor elements" on page 1016
TOTAL_COMMIT_TIME	BIGINT	"total_commit_time - Total commit time monitor element" on page 996

Table 13. Information returned for a unit of work event monitor: Table name: UOW_METRICS (continued). The metrics in this table are the same as those returned in the METRICS monitor element in the UOW_EVENT table

Column Name	Data Type	Description
TOTAL_COMMIT_PROC_TIME	BIGINT	"total_commit_proc_time - Total commits processing time monitor element" on page 995
TOTAL_APP_COMMITS	BIGINT	"total_app_commits - Total application commits monitor elements" on page 990
INT_COMMITS	BIGINT	"int_commits - Internal commits monitor element" on page 698
TOTAL_ROLLBACK_TIME	BIGINT	"total_rollback_time - Total rollback time monitor element" on page 1015
TOTAL_ROLLBACK_PROC_TIME	BIGINT	"total_rollback_proc_time - Total rollback processing time monitor element" on page 1014
TOTAL_APP_ROLLBACKS	BIGINT	"total_app_rollbacks - Total application rollbacks monitor element" on page 991
INT_ROLLBACKS	BIGINT	"int_rollbacks - Internal rollbacks monitor element" on page 700
TOTAL_RUNSTATS_TIME	BIGINT	"total_runstats_time - Total runtime statistics time monitor element" on page 1026
TOTAL_RUNSTATS_PROC_TIME	BIGINT	"total_runstats_proc_time - Total runtime statistics processing time monitor element" on page 1025
TOTAL_RUNSTATS	BIGINT	"total_runstats - Total runtime statistics monitor element" on page 1024
TOTAL_REORG_TIME	BIGINT	"total_reorg_time - Total reorganization time monitor element" on page 1012
TOTAL_REORG_PROC_TIME	BIGINT	"total_reorg_proc_time - Total reorganization processing time monitor element" on page 1011
TOTAL_REORGS	BIGINT	"total_reorgs - Total reorganizations monitor element" on page 1013
TOTAL_LOAD_TIME	BIGINT	"total_load_time - Total load time monitor element" on page 1007
TOTAL_LOAD_PROC_TIME	BIGINT	"total_load_proc_time - Total load processing time monitor element" on page 1006
TOTAL_LOADS	BIGINT	"total_loads - Total loads monitor element" on page 1008
CAT_CACHE_INSERTS	BIGINT	"cat_cache_inserts - Catalog cache inserts monitor element" on page 547
CAT_CACHE_LOOKUPS	BIGINT	"cat_cache_lookups - Catalog cache lookups monitor element" on page 548
PKG_CACHE_INSERTS	BIGINT	"pkg_cache_inserts - Package cache inserts monitor element" on page 801
PKG_CACHE_LOOKUPS	BIGINT	"pkg_cache_lookups - Package cache lookups monitor element" on page 802

Table 13. Information returned for a unit of work event monitor: Table name: UOW_METRICS (continued). The metrics in this table are the same as those returned in the METRICS monitor element in the UOW_EVENT table

Column Name	Data Type	Description
THRESH_VIOLATIONS	BIGINT	"thresh_violations - Number of threshold violations monitor element" on page 981
NUM_LW_THRESH_EXCEEDED	BIGINT	"num_lw_thresh_exceeded - Number of lock wait thresholds exceeded monitor element" on page 778
FCM_TQ_RECV_WAIT_TIME	BIGINT	"fcm_tq_recv_wait_time - FCM table queue received wait time monitor element" on page 656
FCM_MESSAGE_RECV_WAIT_TIME	BIGINT	"fcm_message_recv_wait_time - FCM message received wait time monitor element" on page 641
FCM_TQ_SEND_WAIT_TIME	BIGINT	"fcm_tq_send_wait_time - FCM table queue send wait time monitor element" on page 659
FCM_MESSAGE_SEND_WAIT_TIME	BIGINT	"fcm_message_send_wait_time - FCM message send wait time monitor element" on page 645
AUDIT_FILE_WRITE_WAIT_TIME	BIGINT	"audit_file_write_wait_time - Audit file write wait time monitor element" on page 532
AUDIT_FILE_WRITES_TOTAL	BIGINT	"audit_file_writes_total - Total audit files written monitor element" on page 533
AUDIT_SUBSYSTEM_WAIT_TIME	BIGINT	"audit_subsystem_wait_time - Audit subsystem wait time monitor element" on page 534
AUDIT_SUBSYSTEM_WAITS_TOTAL	BIGINT	"audit_subsystem_waits_total - Total audit subsystem waits monitor element" on page 536
DIAGLOG_WRITE_WAIT_TIME	BIGINT	"diaglog_write_wait_time - Diagnostic log file write wait time monitor element" on page 618
DIAGLOG_WRITES_TOTAL	BIGINT	"diaglog_writes_total - Total diagnostic log file writes monitor element" on page 619
FCM_MESSAGE_SENDS_TOTAL	BIGINT	"fcm_message_sends_total - Total FCM message sends monitor element" on page 646
FCM_MESSAGE_RECVS_TOTAL	BIGINT	"fcm_message_recvs_total - Total FCM message receives monitor element" on page 643
FCM_MESSAGE_SEND_VOLUME	BIGINT	"fcm_message_send_volume - FCM message send volume monitor element" on page 644
FCM_MESSAGE_RECV_VOLUME	BIGINT	"fcm_message_recv_volume - FCM message received volume monitor element" on page 640
FCM_TQ_SENDS_TOTAL	BIGINT	"fcm_tq_sends_total - FCM table queue send total monitor element" on page 660

Table 13. Information returned for a unit of work event monitor: Table name: UOW_METRICS (continued). The metrics in this table are the same as those returned in the METRICS monitor element in the UOW_EVENT table

Column Name	Data Type	Description
FCM_TQ_RECVS_TOTAL	BIGINT	"fcm_tq_recvs_total - FCM table queue receives total monitor element" on page 657
FCM_TQ_SEND_VOLUME	BIGINT	"fcm_tq_send_volume - FCM table queue send volume monitor element" on page 658
FCM_TQ_RECV_VOLUME	BIGINT	"fcm_tq_recv_volume - FCM table queue received volume monitor element" on page 655
TQ_TOT_SEND_SPILLS	BIGINT	"tq_tot_send_spills - Total number of table queue buffers overflowed monitor element" on page 1044
AUDIT_EVENTS_TOTAL	BIGINT	"audit_events_total - Total audit events monitor element" on page 531
TOTAL_ROUTINE_USER_CODE_PROC_TIME	BIGINT	"total_routine_user_code_proc_time - Total routine user code processing time monitor element" on page 1020
TOTAL_ROUTINE_USER_CODE_TIME	BIGINT	"total_routine_user_code_time - Total routine user code time monitor element" on page 1021

Capturing system metrics using the statistics event monitor

The statistics event monitor contains the metrics and the details_xml monitor elements in the event_scstats and event_wlstats logical data groups. Use these monitor elements to capture information about the system in XML documents. Alternatively, view the elements contained in the event_scmetrics and the event_wlmetrics groups.

The monitor elements **metrics** and **details_xml** are XML documents containing all the system monitor elements reported by the MON_GET_SERVICE_SUBCLASS_DETAILS and MON_GET_WORKLOAD_DETAILS table functions. System monitor elements are a subset of the details document reported in the DETAILS column of the MON_GET_SERVICE_SUBCLASS_DETAILS and MON_GET_WORKLOAD_DETAILS table functions.

The monitor elements collected in the XML document **metrics** are reset to 0 after each monitoring interval. The elements in **details_xml** are the same monitor elements, however, they are not reset after each monitoring level. Instead, they continue to increase until the database is deactivated; when the database is reactivated, they are reset to 0.

Important: Starting with Version 9.7 Fix Pack 6, the XML document details_xml is deprecated in the statistics event monitor, and might be removed in a future release. For more information, see Reporting of metrics in details_xml by the statistics event monitor has been deprecated"Reporting of metrics in details_xml by the statistics event monitor has been deprecated" in *What's New for DB2 Version 9.7*.

Request monitor elements are controlled through the COLLECT REQUEST METRICS clause on service superclasses and the mon_req_metrics database

configuration parameter at the database level. Monitor elements are only collected for a request if the request is processed by an agent in a service subclass whose parent service superclass has request monitor element collection enabled, or if request monitor element collection is enabled for the entire database. If request monitor element have been disabled at the database level, and for a service superclass, the metrics reported in the METRICS or DETAILS_XML documents stop increasing (or remain at 0 if request metrics were disabled at database activation time).

The schema for the XML document that is returned in the METRICS and DETAILS_XML columns is available in the file `sqllib/misc/DB2MonCommon.xsd`. The top level element is system_metrics.

Information written to XML for system_metrics and activity_metrics monitor elements

The **`activity_metrics`** monitor element is reported in the MON_GET_ACTIVITY_DETAILS table function, the MON_GET_PKG_CACHE_STMT_DETAILS table function, and the activity event monitor. The **`system_metrics`** monitor element is reported in the MON_GET_CONNECTION_DETAILS, MON_GET_UNIT_OF_WORK_DETAILS, MON_GET_SERVICE_SUBCLASS_DETAILS, MON_GET_WORKLOAD_DETAILS table functions, and the statistics event monitor. This is also documented in the `sqllib/misc/DB2MonCommon.xsd` file.

system_metrics

System level metrics.

Element content: (wlm_queue_time_total , wlm_queue_assignments_total , fcm_tq_recv_wait_time , fcm_message_recv_wait_time , fcm_tq_send_wait_time , fcm_message_send_wait_time , agent_wait_time , agent_waits_total , lock_wait_time , lock_waits , direct_read_time , direct_read_reqs , direct_write_time , direct_write_reqs , log_buffer_wait_time , num_log_buffer_full , log_disk_wait_time , log_disk_waits_total , tcpip_recv_wait_time , tcpip_recvs_total , client_idle_wait_time , ipc_recv_wait_time , ipc_recvs_total , ipc_send_wait_time , ipc_sends_total , tcpip_send_wait_time , tcpip_sends_total , pool_write_time , pool_read_time , audit_file_write_wait_time , audit_file_writes_total , audit_subsystem_wait_time , audit_subsystem_waits_total , diaglog_write_wait_time , diaglog_writes_total , fcm_send_wait_time , fcm_recv_wait_time , total_wait_time , total_rqst_time , rqsts_completed_total , total_app_rqst_time , app_rqsts_completed_total , total_section_sort_proc_time , total_section_sort_time , total_section_sorts , rows_read , rows_modified , pool_data_l_reads , pool_index_l_reads , pool_temp_data_l_reads , pool_temp_index_l_reads , pool_xda_l_reads , pool_temp_xda_l_reads , total_cpu_time , act_completed_total , pool_data_p_reads , pool_temp_data_p_reads , pool_xda_p_reads , pool_temp_xda_p_reads , pool_index_p_reads , pool_temp_index_p_reads , pool_data_writes , pool_xda_writes , pool_index_writes , direct_reads , direct_writes , rows_returned , deadlocks , lock_timeouts , lock_escals , fcm_sends_total , fcm_recvs_total , fcm_send_volume , fcm_recv_volume , fcm_message_sends_total , fcm_message_recvs_total , fcm_message_send_volume , fcm_message_recv_volume , fcm_tq_sends_total , fcm_tq_recvs_total , fcm_tq_send_volume , fcm_tq_recv_volume , tq_tot_send_spills , tcpip_send_volume , tcpip_recv_volume , ipc_send_volume , ipc_recv_volume , post_threshold_sorts , post_shrthreshold_sorts , sort_overflows , audit_events_total , total_rqst_mapped_in {zero or one times (?)}, total_rqst_mapped_out {zero or one

times (?)} , act_rejected_total, act_aborted_total , total_sorts , total_routine_time , total_compile_proc_time , total_compile_time , total_compilations , total_implicit_compile_proc_time , total_implicit_compile_time , total_implicit_compilations , total_runstats_proc_time , total_runstats_time , total_runstats , total_reorg_proc_time , total_reorg_time , total_reorgs , total_load_proc_time , total_load_time , total_loads , total_section_proc_time , total_section_time , total_app_section_executions , total_commit_proc_time , total_commit_time , total_app_commits , total_rollback_proc_time , total_rollback_time , total_app_rollbacks , total_routine_user_code_proc_time , total_routine_user_code_time , thresh_violations , num_lw_thresh_exceeded , total_routine_invocations , int_commits , int_rollbacks , cat_cache_inserts , cat_cache_lookups , pkg_cache_inserts , pkg_cache_lookups , act_rqsts_total , total_act_wait_time , total_act_time , ANY content (skip) {zero or more (*)})

Attributes:

QName	Type	Fixed	Default	Use	Annotation
release	xs:long			required	
ANY attribute from ANY namespace					

activity_metrics

Activity level metrics.

Element content: (wlm_queue_time_total , wlm_queue_assignments_total , fcm_tq_recv_wait_time , fcm_message_recv_wait_time , fcm_tq_send_wait_time , fcm_message_send_wait_time , lock_wait_time , lock_waits , direct_read_time , direct_read_reqs , direct_write_time , direct_write_reqs , log_buffer_wait_time , num_log_buffer_full , log_disk_wait_time , log_disk_waits_total , pool_write_time , pool_read_time , audit_file_write_wait_time , audit_file_writes_total , audit_subsystem_wait_time , audit_subsystem_waits_total , diaglog_write_wait_time , diaglog_writes_total , fcm_send_wait_time , fcm_recv_wait_time , total_act_wait_time , total_section_sort_proc_time , total_section_sort_time , total_section_sorts , total_act_time , rows_read , rows_modified , pool_data_l_reads , pool_index_l_reads , pool_temp_data_l_reads , pool_temp_index_l_reads , pool_xda_l_reads , pool_temp_xda_l_reads , total_cpu_time , pool_data_p_reads , pool_temp_data_p_reads , pool_xda_p_reads , pool_temp_xda_p_reads , pool_index_p_reads , pool_temp_index_p_reads , pool_data_writes , pool_xda_writes , pool_index_writes , direct_reads , direct_writes , rows_returned , deadlocks , lock_timeouts , lock_escals , fcm_sends_total , fcm_recvs_total , fcm_send_volume , fcm_recv_volume , fcm_message_sends_total , fcm_message_recvs_total , fcm_message_send_volume , fcm_message_recv_volume , fcm_tq_sends_total , fcm_tq_recvs_total , fcm_tq_send_volume , fcm_tq_recv_volume , tq_tot_send_spills , post_threshold_sorts , post_shrthreshold_sorts , sort_overflows , audit_events_total , total_sorts , stmt_exec_time , coord_stmt_exec_time {zero or one times (?)} , total_routine_non_sect_proc_time , total_routine_non_sect_time , total_section_proc_time , total_section_time , total_app_section_executions , total_routine_user_code_proc_time , total_routine_user_code_time , total_routine_time , thresh_violations , num_lw_thresh_exceeded , total_routine_invocations , ANY content (skip) {zero or more (*)})

Attributes:

QName	Type	Fixed	Default	Use	Annotation
release	xs:long			required	
ANY attribute from ANY namespace					

agent_wait_time

See monitor element "agent_wait_time - Agent wait time monitor element" on page 511 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

agent_waits_total

See monitor element "agent_waits_total - Total agent waits monitor element" on page 512 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

tcpip_recv_wait_time

See monitor element "tcpip_recv_wait_time - TCP/IP received wait time monitor element" on page 976 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

tcpip_recvs_total

See monitor element "tcpip_recvs_total - TCP/IP receives total monitor element" on page 977 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

client_idle_wait_time

See monitor element "client_idle_wait_time - Client idle wait time monitor element" on page 557 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

ipc_recv_wait_time

See monitor element "ipc_recv_wait_time - Interprocess communication received wait time monitor element" on page 705 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

ipc_recvs_total

See monitor element "ipc_recvs_total - Interprocess communication receives total monitor element" on page 706 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

ipc_send_wait_time

See monitor element "ipc_send_wait_time - Interprocess communication send wait time monitor element" on page 708 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

ipc_sends_total

See monitor element "ipc_sends_total - Interprocess communication send total monitor element" on page 709 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

tcpip_send_wait_time

See monitor element "tcpip_send_wait_time - TCP/IP send wait time monitor element" on page 979 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

tcpip_sends_total

See monitor element "tcpip_sends_total - TCP/IP sends total monitor element" on page 980 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_wait_time

See monitor element "total_wait_time - Total wait time monitor element" on page 1038 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_rqst_time

See monitor element "total_rqst_time - Total request time monitor element" on page 1023 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

rqsts_completed_total

See monitor element “rqsts_completed_total - Total requests completed monitor element” on page 897 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_app_rqst_time

See monitor element “total_app_rqst_time - Total application request time monitor element” on page 992 for more details.

Contained by:system_metrics

Element content:

Type	Facet
xs:long	

app_rqsts_completed_total

See monitor element “app_rqsts_completed_total - Total application requests completed monitor element” on page 518 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

act_completed_total

See monitor element “act_completed_total - Total completed activities monitor element” on page 495 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

tcpip_send_volume

See monitor element “tcpip_send_volume - TCP/IP send volume monitor element” on page 978 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

tcpip_recv_volume

See monitor element “tcpip_recv_volume - TCP/IP received volume monitor element” on page 975 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

ipc_send_volume

See monitor element “ipc_send_volume - Interprocess communication send volume monitor element” on page 707 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

ipc_recv_volume

See monitor element “ipc_recv_volume - Interprocess communication received volume monitor element” on page 704 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_rqst_mapped_in

See monitor element "total_rqst_mapped_in - Total request mapped-in monitor element" on page 1022 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_rqst_mapped_out

See monitor element "total_rqst_mapped_out - Total request mapped-out monitor element" on page 1022 for more details.

Contained by:system_metrics

Element content:

Type	Facet
xs:long	

act_rejected_total

See monitor element "act_rejected_total - Total rejected activities monitor element" on page 497 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

act_aborted_total

See monitor element "act_aborted_total - Total aborted activities monitor element" on page 494 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_compile_proc_time

See monitor element "total_compile_proc_time - Total compile processing time monitor element" on page 998 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_compile_time

See monitor element "total_compile_time - Total compile time monitor element" on page 999 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_compilations

See monitor element "total_compilations - Total compilations monitor element" on page 997 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_implicit_compile_proc_time

See monitor element "total_implicit_compile_proc_time - Total implicit compile processing time monitor element" on page 1004 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_implicit_compile_time

See monitor element "total_implicit_compile_time - Total implicit compile time monitor element" on page 1005 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_implicit_compilations

See monitor element "total_implicit_compilations - Total implicit complications monitor element" on page 1003 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_runstats_proc_time

See monitor element "total_runstats_proc_time - Total runtime statistics processing time monitor element" on page 1025 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_runstats_time

See monitor element "total_runstats_time - Total runtime statistics time monitor element" on page 1026 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_runstats

See monitor element "total_runstats - Total runtime statistics monitor element" on page 1024 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_reorg_proc_time

See monitor element "total_reorg_proc_time - Total reorganization processing time monitor element" on page 1011 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_reorg_time

See monitor element "total_reorg_time - Total reorganization time monitor element" on page 1012 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_reorgs

See monitor element "total_reorgs - Total reorganizations monitor element" on page 1013 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_load_proc_time

See monitor element "total_load_proc_time - Total load processing time monitor element" on page 1006 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_load_time

See monitor element "total_load_time - Total load time monitor element" on page 1007 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_loads

See monitor element "total_loads - Total loads monitor element" on page 1008 for more details.

Contained by:system_metrics

Element content:

Type	Facet
xs:long	

total_commit_proc_time

See monitor element "total_commit_proc_time - Total commits processing time monitor element" on page 995 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_commit_time

See monitor element "total_commit_time - Total commit time monitor element" on page 996 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_app_commits

See monitor element "total_app_commits - Total application commits monitor elements" on page 990 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_rollback_proc_time

See monitor element "total_rollback_proc_time - Total rollback processing time monitor element" on page 1014 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_rollback_time

See monitor element "total_rollback_time - Total rollback time monitor element" on page 1015 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_app_rollbacks

See monitor element "total_app_rollbacks - Total application rollbacks monitor element" on page 991 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

int_commits

See monitor element "int_commits - Internal commits monitor element" on page 698 for more details.

Contained by:system_metrics

Element content:

Type	Facet
xs:long	

int_rollbacks

See monitor element "int_rollbacks - Internal rollbacks monitor element" on page 700 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

cat_cache_inserts

See monitor element "cat_cache_inserts - Catalog cache inserts monitor element" on page 547 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

cat_cache_lookups

See monitor element "cat_cache_lookups - Catalog cache lookups monitor element" on page 548 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

pkg_cache_inserts

See monitor element "pkg_cache_inserts - Package cache inserts monitor element" on page 801 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

pkg_cache_lookups

See monitor element "pkg_cache_lookups - Package cache lookups monitor element" on page 802 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

act_rqsts_total

See monitor element "act_rqsts_total - Total activity requests monitor elements" on page 500 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

wlm_queue_time_total

See monitor element "wlm_queue_time_total - Workload manager total queue time monitor element" on page 1060 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

wlm_queue_assignments_total

See monitor element "wlm_queue_assignments_total - Workload manager total queue assignments monitor element" on page 1059 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_tq_recv_wait_time

See monitor element "fcm_tq_recv_wait_time - FCM table queue received wait time monitor element" on page 656 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_message_recv_wait_time

See monitor element "fcm_message_recv_wait_time - FCM message received wait time monitor element" on page 641 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_tq_send_wait_time

See monitor element "fcm_tq_send_wait_time - FCM table queue send wait time monitor element" on page 659 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_message_send_wait_time

See monitor element "fcm_message_send_wait_time - FCM message send wait time monitor element" on page 645 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

lock_wait_time

See monitor element "lock_wait_time - Time waited on locks monitor element" on page 734 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

lock_waits

See monitor element "lock_waits - Lock waits monitor element" on page 737 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

direct_read_time

See monitor element "direct_read_time - Direct read time monitor element" on page 622 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

direct_read_reqs

See monitor element "direct_read_reqs - Direct read requests monitor element" on page 620 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

direct_write_time

See monitor element "direct_write_time - Direct write time monitor element" on page 627 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

direct_write_reqs

See monitor element "direct_write_reqs - Direct write requests monitor element" on page 626 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

log_buffer_wait_time

See monitor element "log_buffer_wait_time - Log buffer wait time monitor element" on page 740 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

num_log_buffer_full

See monitor element "num_log_buffer_full - Number of times full log buffer caused agents to wait monitor element" on page 775 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

log_disk_wait_time

See monitor element "log_disk_wait_time - Log disk wait time monitor element" on page 741 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

log_disk_waits_total

See monitor element "log_disk_waits_total - Total log disk waits monitor element" on page 743 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_write_time

See monitor element "pool_write_time - Total buffer pool physical write time monitor element" on page 847 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_read_time

See monitor element "pool_read_time - Total buffer pool physical read time monitor element" on page 833 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

audit_file_write_wait_time

See monitor element "audit_file_write_wait_time - Audit file write wait time monitor element" on page 532 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

audit_file_writes_total

See monitor element "audit_file_writes_total - Total audit files written monitor element" on page 533 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

audit_subsystem_wait_time

See monitor element "audit_subsystem_wait_time - Audit subsystem wait time monitor element" on page 534 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

audit_subsystem_waits_total

See monitor element "audit_subsystem_waits_total - Total audit subsystem waits monitor element" on page 536 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

diaglog_write_wait_time

See monitor element "diaglog_write_wait_time - Diagnostic log file write wait time monitor element" on page 618 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

diaglog_writes_total

See monitor element "diaglog_writes_total - Total diagnostic log file writes monitor element" on page 619 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_send_wait_time

See monitor element "fcm_send_wait_time - FCM send wait time monitor element" on page 652 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_recv_wait_time

See monitor element “fcm_recv_wait_time - FCM received wait time monitor element” on page 648 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_act_wait_time

See monitor element “total_act_wait_time - Total activity wait time monitor element” on page 989 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_section_sort_proc_time

See monitor element “total_section_sort_proc_time - Total section sort processing time monitor element” on page 1028 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_section_sort_time

See monitor element “total_section_sort_time - Total section sort time monitor element” on page 1030 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_section_sorts

See monitor element "total_section_sorts - Total section sorts monitor element" on page 1032 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_act_time

See monitor element "total_act_time - Total activity time monitor element" on page 988 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

rows_read

See monitor element "rows_read - Rows read monitor element" on page 892 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

rows_modified

See monitor element "rows_modified - Rows modified monitor element" on page 890 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_data_l_reads

See monitor element "pool_data_l_reads - Buffer pool data logical reads monitor element" on page 816 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_index_l_reads

See monitor element "pool_index_l_reads - Buffer pool index logical reads monitor element" on page 825 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_temp_data_l_reads

See monitor element "pool_temp_data_l_reads - Buffer pool temporary data logical reads monitor element" on page 835 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_temp_index_l_reads

See monitor element "pool_temp_index_l_reads - Buffer pool temporary index logical reads monitor element" on page 839 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_xda_l_reads

See monitor element "pool_xda_l_reads - Buffer pool XDA data logical reads monitor element" on page 849 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_temp_xda_l_reads

See monitor element "pool_temp_xda_l_reads - Buffer pool temporary XDA data logical reads monitor element" on page 843 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_cpu_time

See monitor element "total_cpu_time - Total CPU time monitor element" on page 1001 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_data_p_reads

See monitor element "pool_data_p_reads - Buffer pool data physical reads monitor element" on page 818 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_temp_data_p_reads

See monitor element "pool_temp_data_p_reads - Buffer pool temporary data physical reads monitor element" on page 837 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_xda_p_reads

See monitor element "pool_xda_p_reads - Buffer pool XDA data physical reads monitor element" on page 851 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_temp_xda_p_reads

See monitor element "pool_temp_xda_p_reads - Buffer pool temporary XDA data physical reads monitor element" on page 845 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_index_p_reads

See monitor element "pool_index_p_reads - Buffer pool index physical reads monitor element" on page 827 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_temp_index_p_reads

See monitor element "pool_temp_index_p_reads - Buffer pool temporary index physical reads monitor element" on page 841 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_data_writes

See monitor element "pool_data_writes - Buffer pool data writes monitor element" on page 820 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_xda_writes

See monitor element "pool_xda_writes - Buffer pool XDA data writes monitor element" on page 853 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_index_writes

See monitor element "pool_index_writes - Buffer pool index writes monitor element" on page 829 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

direct_reads

See monitor element "direct_reads - Direct reads from database monitor element" on page 624 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

direct_writes

See monitor element "direct_writes - Direct writes to database monitor element" on page 629 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

rows_returned

See monitor element "rows_returned - Rows returned monitor element" on page 893 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

deadlocks

See monitor element "deadlocks - Deadlocks detected monitor element" on page 614 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

lock_timeouts

See monitor element "lock_timeouts - Number of lock timeouts monitor element" on page 732 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

lock_escals

See monitor element "lock_escals - Number of lock escalations monitor element" on page 721 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_sends_total

See monitor element "fcm_sends_total - FCM sends total monitor element" on page 654 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_recvs_total

See monitor element "fcm_recvs_total - FCM receives total monitor element" on page 650 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_send_volume

See monitor element "fcm_send_volume - FCM send volume monitor element" on page 651 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_recv_volume

See monitor element "fcm_recv_volume - FCM received volume monitor element" on page 647 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_message_sends_total

See monitor element "fcm_message_sends_total - Total FCM message sends monitor element" on page 646 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_message_recvs_total

See monitor element "fcm_message_recvs_total - Total FCM message receives monitor element" on page 643 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_message_send_volume

See monitor element "fcm_message_send_volume - FCM message send volume monitor element" on page 644 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_message_recv_volume

See monitor element "fcm_message_recv_volume - FCM message received volume monitor element" on page 640 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_tq_sends_total

See monitor element "fcm_tq_sends_total - FCM table queue send total monitor element" on page 660 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_tq_recvs_total

See monitor element “fcm_tq_recvs_total - FCM table queue receives total monitor element” on page 657 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_tq_send_volume

See monitor element “fcm_tq_send_volume - FCM table queue send volume monitor element” on page 658 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_tq_recv_volume

See monitor element “fcm_tq_recv_volume - FCM table queue received volume monitor element” on page 655 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

tq_tot_send_spills

See monitor element “tq_tot_send_spills - Total number of table queue buffers overflowed monitor element” on page 1044 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

post_threshold_sorts

See monitor element "post_threshold_sorts - Post threshold sorts monitor element" on page 858 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

post_shrthreshold_sorts

See monitor element "post_shrthreshold_sorts - Post shared threshold sorts monitor element" on page 856 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

sort_overflows

See monitor element "sort_overflows - Sort overflows monitor element" on page 915 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

audit_events_total

See monitor element "audit_events_total - Total audit events monitor element" on page 531 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_sorts

See monitor element "total_sorts - Total sorts monitor element" on page 1036 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

stmt_exec_time

See monitor element "stmt_exec_time - Statement execution time monitor element" on page 931 for more details.

Contained by: activity_metrics

Element content:

Type	Facet
xs:long	

coord_stmt_exec_time

See monitor element "coord_stmt_exec_time - Execution time for statement by coordinator agent monitor element" on page 591 for more details.

Contained by: activity_metrics

Element content:

Type	Facet
xs:long	

total_routine_non_sect_proc_time

See monitor element "total_routine_non_sect_proc_time - Non-section processing time monitor element" on page 1017 for more details.

Contained by: activity_metrics

Element content:

Type	Facet
xs:long	

total_routine_non_sect_time

See monitor element "total_routine_non_sect_time - Non-section routine execution time monitor elements" on page 1017 for more details.

Contained by: activity_metrics

Element content:

Type	Facet
xs:long	

total_section_proc_time

See monitor element "total_section_proc_time - Total section processing time monitor element" on page 1027 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_section_time

See monitor element "total_section_time - Total section time monitor element" on page 1033 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_app_section_executions

See monitor element "total_app_section_executions - Total application section executions monitor element" on page 993 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_routine_user_code_proc_time

See monitor element "total_routine_user_code_proc_time - Total routine user code processing time monitor element" on page 1020 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_routine_user_code_time

See monitor element "total_routine_user_code_time - Total routine user code time monitor element" on page 1021 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_routine_time

See monitor element "total_routine_time - Total routine time monitor element" on page 1018 for more details.

Contained by: system_metricsactivity_metrics

Element content:

Type	Facet
xs:long	

thresh_violations

See monitor element "thresh_violations - Number of threshold violations monitor element" on page 981 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

num_lw_thresh_exceeded

See monitor element "num_lw_thresh_exceeded - Number of lock wait thresholds exceeded monitor element" on page 778 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_routine_invocations

See monitor element "total_routine_invocations - Total routine invocations monitor elements" on page 1016 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

Capturing activity monitor elements using the activity event monitor

The activity event monitor contains the details_xml monitor element in the event_activity logical data groups. Use this monitor element to capture information about activities.

The monitor element **details_xml** returns an XML document containing all the activity monitor elements reported by the MON_GET_ACTIVITY_DETAILS table function. Activity monitor elements are a subset of the activity details document reported in the DETAILS column of the MON_GET_ACTIVITY_DETAILS table function.

Activity monitor elements are controlled through the COLLECT ACTIVITY METRICS clause on workloads, or the **mon_act_metrics** database configuration parameter at the database level. Monitor elements are collected if the connection that submits the activity is associated with a workload or database for which activity monitor element collection is enabled. If activity monitor element are not collected for an activity, DETAILS_XML contains an empty document.

The schema for the XML document that is returned in the DETAILS_XML column is available in the file `sqllib/misc/DB2MonCommon.xsd`. The top level element is activity_metrics.

Information written to XML for system_metrics and activity_metrics monitor elements

The **activity_metrics** monitor element is reported in the MON_GET_ACTIVITY_DETAILS table function, the MON_GET_PKG_CACHE_STMT_DETAILS table function, and the activity event monitor. The **system_metrics** monitor element is reported in the MON_GET_CONNECTION_DETAILS, MON_GET_UNIT_OF_WORK_DETAILS, MON_GET_SERVICE_SUBCLASS_DETAILS, MON_GET_WORKLOAD_DETAILS table functions, and the statistics event monitor. This is also documented in the `sqllib/misc/DB2MonCommon.xsd` file.

system_metrics

System level metrics.

Element content: (wlm_queue_time_total , wlm_queue_assignments_total , fcm_tq_recv_wait_time , fcm_message_recv_wait_time , fcm_tq_send_wait_time , fcm_message_send_wait_time , agent_wait_time , agent_waits_total , lock_wait_time , lock_waits , direct_read_time , direct_read_reqs , direct_write_time

, direct_write_reqs , log_buffer_wait_time , num_log_buffer_full ,
log_disk_wait_time , log_disk_waits_total , tcpip_recv_wait_time , tcpip_recvs_total
, client_idle_wait_time , ipc_recv_wait_time , ipc_recvs_total , ipc_send_wait_time ,
ipc_sends_total , tcpip_send_wait_time , tcpip_sends_total , pool_write_time ,
pool_read_time , audit_file_write_wait_time , audit_file_writes_total ,
audit_subsystem_wait_time , audit_subsystem_waits_total ,
diaglog_write_wait_time , diaglog_writes_total , fcm_send_wait_time ,
fcm_recv_wait_time , total_wait_time , total_rqst_time , rqsts_completed_total ,
total_app_rqst_time , app_rqsts_completed_total , total_section_sort_proc_time ,
total_section_sort_time , total_section_sorts , rows_read , rows_modified ,
pool_data_l_reads , pool_index_l_reads , pool_temp_data_l_reads ,
pool_temp_index_l_reads , pool_xda_l_reads , pool_temp_xda_l_reads ,
total_cpu_time , act_completed_total , pool_data_p_reads ,
pool_temp_data_p_reads , pool_xda_p_reads , pool_temp_xda_p_reads ,
pool_index_p_reads , pool_temp_index_p_reads , pool_data_writes ,
pool_xda_writes , pool_index_writes , direct_reads , direct_writes , rows_returned ,
deadlocks , lock_timeouts , lock_escals , fcm_sends_total , fcm_recvs_total ,
fcm_send_volume , fcm_recv_volume , fcm_message_sends_total ,
fcm_message_recvs_total , fcm_message_send_volume , fcm_message_recv_volume
, fcm_tq_sends_total , fcm_tq_recvs_total , fcm_tq_send_volume ,
fcm_tq_recv_volume , tq_tot_send_spills , tcpip_send_volume , tcpip_recv_volume
, ipc_send_volume , ipc_recv_volume , post_threshold_sorts ,
post_shrthreshold_sorts , sort_overflows , audit_events_total ,
total_rqst_mapped_in {zero or one times (?)}, total_rqst_mapped_out {zero or one
times (?)} , act_rejected_total, act_aborted_total , total_sorts , total_routine_time ,
total_compile_proc_time , total_compile_time , total_compilations ,
total_implicit_compile_proc_time , total_implicit_compile_time ,
total_implicit_compilations , total_runstats_proc_time , total_runstats_time ,
total_runstats , total_reorg_proc_time , total_reorg_time , total_reorgs ,
total_load_proc_time , total_load_time , total_loads , total_section_proc_time ,
total_section_time , total_app_section_executions , total_commit_proc_time ,
total_commit_time , total_app_commits , total_rollback_proc_time ,
total_rollback_time , total_app_rollbacks , total_routine_user_code_proc_time ,
total_routine_user_code_time , thresh_violations , num_lw_thresh_exceeded ,
total_routine_invocations , int_commits , int_rollbacks , cat_cache_inserts ,
cat_cache_lookups , pkg_cache_inserts , pkg_cache_lookups , act_rqsts_total ,
total_act_wait_time , total_act_time , ANY content (skip) {zero or more (*)})

Attributes:

QName	Type	Fixed	Default	Use	Annotation
release	xs:long			required	
ANY attribute from ANY namespace					

activity_metrics

Activity level metrics.

Element content: (wlm_queue_time_total , wlm_queue_assignments_total ,
fcm_tq_recv_wait_time , fcm_message_recv_wait_time , fcm_tq_send_wait_time ,
fcm_message_send_wait_time , lock_wait_time , lock_waits , direct_read_time ,
direct_read_reqs , direct_write_time , direct_write_reqs , log_buffer_wait_time ,
num_log_buffer_full , log_disk_wait_time , log_disk_waits_total , pool_write_time ,

pool_read_time , audit_file_write_wait_time , audit_file_writes_total , audit_subsystem_wait_time , audit_subsystem_waits_total , diaglog_write_wait_time , diaglog_writes_total , fcm_send_wait_time , fcm_recv_wait_time , total_act_wait_time , total_section_sort_proc_time , total_section_sort_time , total_section_sorts , total_act_time , rows_read , rows_modified , pool_data_l_reads , pool_index_l_reads , pool_temp_data_l_reads , pool_temp_index_l_reads , pool_xda_l_reads , pool_temp_xda_l_reads , total_cpu_time , pool_data_p_reads , pool_temp_data_p_reads , pool_xda_p_reads , pool_temp_xda_p_reads , pool_index_p_reads , pool_temp_index_p_reads , pool_data_writes , pool_xda_writes , pool_index_writes , direct_reads , direct_writes , rows_returned , deadlocks , lock_timeouts , lock_escals , fcm_sends_total , fcm_recvs_total , fcm_send_volume , fcm_recv_volume , fcm_message_sends_total , fcm_message_recvs_total , fcm_message_send_volume , fcm_message_recv_volume , fcm_tq_sends_total , fcm_tq_recvs_total , fcm_tq_send_volume , fcm_tq_recv_volume , tq_tot_send_spills , post_threshold_sorts , post_shrthreshold_sorts , sort_overflows , audit_events_total , total_sorts , stmt_exec_time , coord_stmt_exec_time {zero or one times (?)} , total_routine_non_sect_proc_time , total_routine_non_sect_time , total_section_proc_time , total_section_time , total_app_section_executions , total_routine_user_code_proc_time , total_routine_user_code_time , total_routine_time , thresh_violations , num_lw_thresh_exceeded , total_routine_invocations , ANY content (skip) {zero or more (*)})

Attributes:

QName	Type	Fixed	Default	Use	Annotation
release	xs:long			required	
ANY attribute from ANY namespace					

agent_wait_time

See monitor element “agent_wait_time - Agent wait time monitor element” on page 511 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

agent_waits_total

See monitor element “agent_waits_total - Total agent waits monitor element” on page 512 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

tcpip_recv_wait_time

See monitor element "tcpip_recv_wait_time - TCP/IP received wait time monitor element" on page 976 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

tcpip_recvs_total

See monitor element "tcpip_recvs_total - TCP/IP receives total monitor element" on page 977 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

client_idle_wait_time

See monitor element "client_idle_wait_time - Client idle wait time monitor element" on page 557 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

ipc_recv_wait_time

See monitor element "ipc_recv_wait_time - Interprocess communication received wait time monitor element" on page 705 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

ipc_recvs_total

See monitor element "ipc_recvs_total - Interprocess communication receives total monitor element" on page 706 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

ipc_send_wait_time

See monitor element "ipc_send_wait_time - Interprocess communication send wait time monitor element" on page 708 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

ipc_sends_total

See monitor element "ipc_sends_total - Interprocess communication send total monitor element" on page 709 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

tcpip_send_wait_time

See monitor element "tcpip_send_wait_time - TCP/IP send wait time monitor element" on page 979 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

tcpip_sends_total

See monitor element "tcpip_sends_total - TCP/IP sends total monitor element" on page 980 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_wait_time

See monitor element "total_wait_time - Total wait time monitor element" on page 1038 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_rqst_time

See monitor element "total_rqst_time - Total request time monitor element" on page 1023 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

rqsts_completed_total

See monitor element "rqsts_completed_total - Total requests completed monitor element" on page 897 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_app_rqst_time

See monitor element "total_app_rqst_time - Total application request time monitor element" on page 992 for more details.

Contained by:system_metrics

Element content:

Type	Facet
xs:long	

app_rqsts_completed_total

See monitor element "app_rqsts_completed_total - Total application requests completed monitor element" on page 518 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

act_completed_total

See monitor element "act_completed_total - Total completed activities monitor element" on page 495 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

tcpip_send_volume

See monitor element "tcpip_send_volume - TCP/IP send volume monitor element" on page 978 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

tcpip_recv_volume

See monitor element "tcpip_recv_volume - TCP/IP received volume monitor element" on page 975 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

ipc_send_volume

See monitor element "ipc_send_volume - Interprocess communication send volume monitor element" on page 707 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

ipc_recv_volume

See monitor element "ipc_recv_volume - Interprocess communication received volume monitor element" on page 704 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_rqst_mapped_in

See monitor element "total_rqst_mapped_in - Total request mapped-in monitor element" on page 1022 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_rqst_mapped_out

See monitor element "total_rqst_mapped_out - Total request mapped-out monitor element" on page 1022 for more details.

Contained by:system_metrics

Element content:

Type	Facet
xs:long	

act_rejected_total

See monitor element "act_rejected_total - Total rejected activities monitor element" on page 497 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

act_aborted_total

See monitor element "act_aborted_total - Total aborted activities monitor element" on page 494 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_compile_proc_time

See monitor element "total_compile_proc_time - Total compile processing time monitor element" on page 998 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_compile_time

See monitor element "total_compile_time - Total compile time monitor element" on page 999 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_compilations

See monitor element "total_compilations - Total compilations monitor element" on page 997 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_implicit_compile_proc_time

See monitor element "total_implicit_compile_proc_time - Total implicit compile processing time monitor element" on page 1004 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_implicit_compile_time

See monitor element "total_implicit_compile_time - Total implicit compile time monitor element" on page 1005 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_implicit_compilations

See monitor element "total_implicit_compilations - Total implicit complications monitor element" on page 1003 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_runstats_proc_time

See monitor element "total_runstats_proc_time - Total runtime statistics processing time monitor element" on page 1025 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_runstats_time

See monitor element "total_runstats_time - Total runtime statistics time monitor element" on page 1026 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_runstats

See monitor element "total_runstats - Total runtime statistics monitor element" on page 1024 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_reorg_proc_time

See monitor element "total_reorg_proc_time - Total reorganization processing time monitor element" on page 1011 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_reorg_time

See monitor element "total_reorg_time - Total reorganization time monitor element" on page 1012 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_reorgs

See monitor element "total_reorgs - Total reorganizations monitor element" on page 1013 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_load_proc_time

See monitor element "total_load_proc_time - Total load processing time monitor element" on page 1006 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_load_time

See monitor element "total_load_time - Total load time monitor element" on page 1007 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_loads

See monitor element "total_loads - Total loads monitor element" on page 1008 for more details.

Contained by:system_metrics

Element content:

Type	Facet
xs:long	

total_commit_proc_time

See monitor element "total_commit_proc_time - Total commits processing time monitor element" on page 995 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_commit_time

See monitor element "total_commit_time - Total commit time monitor element" on page 996 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_app_commits

See monitor element "total_app_commits - Total application commits monitor elements" on page 990 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_rollback_proc_time

See monitor element "total_rollback_proc_time - Total rollback processing time monitor element" on page 1014 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_rollback_time

See monitor element "total_rollback_time - Total rollback time monitor element" on page 1015 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

total_app_rollbacks

See monitor element "total_app_rollbacks - Total application rollbacks monitor element" on page 991 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

int_commits

See monitor element "int_commits - Internal commits monitor element" on page 698 for more details.

Contained by:system_metrics

Element content:

Type	Facet
xs:long	

int_rollbacks

See monitor element "int_rollbacks - Internal rollbacks monitor element" on page 700 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

cat_cache_inserts

See monitor element "cat_cache_inserts - Catalog cache inserts monitor element" on page 547 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

cat_cache_lookups

See monitor element "cat_cache_lookups - Catalog cache lookups monitor element" on page 548 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

pkg_cache_inserts

See monitor element "pkg_cache_inserts - Package cache inserts monitor element" on page 801 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

pkg_cache_lookups

See monitor element "pkg_cache_lookups - Package cache lookups monitor element" on page 802 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

act_rqsts_total

See monitor element "act_rqsts_total - Total activity requests monitor elements" on page 500 for more details.

Contained by: system_metrics

Element content:

Type	Facet
xs:long	

wlm_queue_time_total

See monitor element "wlm_queue_time_total - Workload manager total queue time monitor element" on page 1060 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

wlm_queue_assignments_total

See monitor element "wlm_queue_assignments_total - Workload manager total queue assignments monitor element" on page 1059 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_tq_recv_wait_time

See monitor element "fcm_tq_recv_wait_time - FCM table queue received wait time monitor element" on page 656 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_message_recv_wait_time

See monitor element "fcm_message_recv_wait_time - FCM message received wait time monitor element" on page 641 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_tq_send_wait_time

See monitor element "fcm_tq_send_wait_time - FCM table queue send wait time monitor element" on page 659 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_message_send_wait_time

See monitor element "fcm_message_send_wait_time - FCM message send wait time monitor element" on page 645 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

lock_wait_time

See monitor element "lock_wait_time - Time waited on locks monitor element" on page 734 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

lock_waits

See monitor element "lock_waits - Lock waits monitor element" on page 737 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

direct_read_time

See monitor element "direct_read_time - Direct read time monitor element" on page 622 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

direct_read_reqs

See monitor element "direct_read_reqs - Direct read requests monitor element" on page 620 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

direct_write_time

See monitor element "direct_write_time - Direct write time monitor element" on page 627 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

direct_write_reqs

See monitor element "direct_write_reqs - Direct write requests monitor element" on page 626 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

log_buffer_wait_time

See monitor element "log_buffer_wait_time - Log buffer wait time monitor element" on page 740 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

num_log_buffer_full

See monitor element "num_log_buffer_full - Number of times full log buffer caused agents to wait monitor element" on page 775 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

log_disk_wait_time

See monitor element "log_disk_wait_time - Log disk wait time monitor element" on page 741 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

log_disk_waits_total

See monitor element "log_disk_waits_total - Total log disk waits monitor element" on page 743 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_write_time

See monitor element "pool_write_time - Total buffer pool physical write time monitor element" on page 847 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_read_time

See monitor element "pool_read_time - Total buffer pool physical read time monitor element" on page 833 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

audit_file_write_wait_time

See monitor element "audit_file_write_wait_time - Audit file write wait time monitor element" on page 532 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

audit_file_writes_total

See monitor element "audit_file_writes_total - Total audit files written monitor element" on page 533 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

audit_subsystem_wait_time

See monitor element "audit_subsystem_wait_time - Audit subsystem wait time monitor element" on page 534 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

audit_subsystem_waits_total

See monitor element "audit_subsystem_waits_total - Total audit subsystem waits monitor element" on page 536 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

diaglog_write_wait_time

See monitor element "diaglog_write_wait_time - Diagnostic log file write wait time monitor element" on page 618 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

diaglog_writes_total

See monitor element “diaglog_writes_total - Total diagnostic log file writes monitor element” on page 619 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_send_wait_time

See monitor element “fcm_send_wait_time - FCM send wait time monitor element” on page 652 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_recv_wait_time

See monitor element “fcm_recv_wait_time - FCM received wait time monitor element” on page 648 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_act_wait_time

See monitor element “total_act_wait_time - Total activity wait time monitor element” on page 989 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_section_sort_proc_time

See monitor element "total_section_sort_proc_time - Total section sort processing time monitor element" on page 1028 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_section_sort_time

See monitor element "total_section_sort_time - Total section sort time monitor element" on page 1030 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_section_sorts

See monitor element "total_section_sorts - Total section sorts monitor element" on page 1032 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_act_time

See monitor element "total_act_time - Total activity time monitor element" on page 988 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

rows_read

See monitor element "rows_read - Rows read monitor element" on page 892 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

rows_modified

See monitor element "rows_modified - Rows modified monitor element" on page 890 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_data_l_reads

See monitor element "pool_data_l_reads - Buffer pool data logical reads monitor element" on page 816 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_index_l_reads

See monitor element "pool_index_l_reads - Buffer pool index logical reads monitor element" on page 825 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_temp_data_l_reads

See monitor element "pool_temp_data_l_reads - Buffer pool temporary data logical reads monitor element" on page 835 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_temp_index_l_reads

See monitor element "pool_temp_index_l_reads - Buffer pool temporary index logical reads monitor element" on page 839 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_xda_l_reads

See monitor element "pool_xda_l_reads - Buffer pool XDA data logical reads monitor element" on page 849 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_temp_xda_l_reads

See monitor element "pool_temp_xda_l_reads - Buffer pool temporary XDA data logical reads monitor element" on page 843 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_cpu_time

See monitor element "total_cpu_time - Total CPU time monitor element" on page 1001 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_data_p_reads

See monitor element "pool_data_p_reads - Buffer pool data physical reads monitor element" on page 818 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_temp_data_p_reads

See monitor element "pool_temp_data_p_reads - Buffer pool temporary data physical reads monitor element" on page 837 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_xda_p_reads

See monitor element "pool_xda_p_reads - Buffer pool XDA data physical reads monitor element" on page 851 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_temp_xda_p_reads

See monitor element "pool_temp_xda_p_reads - Buffer pool temporary XDA data physical reads monitor element" on page 845 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_index_p_reads

See monitor element "pool_index_p_reads - Buffer pool index physical reads monitor element" on page 827 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_temp_index_p_reads

See monitor element "pool_temp_index_p_reads - Buffer pool temporary index physical reads monitor element" on page 841 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_data_writes

See monitor element "pool_data_writes - Buffer pool data writes monitor element" on page 820 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_xda_writes

See monitor element "pool_xda_writes - Buffer pool XDA data writes monitor element" on page 853 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

pool_index_writes

See monitor element "pool_index_writes - Buffer pool index writes monitor element" on page 829 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

direct_reads

See monitor element "direct_reads - Direct reads from database monitor element" on page 624 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

direct_writes

See monitor element "direct_writes - Direct writes to database monitor element" on page 629 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

rows_returned

See monitor element "rows_returned - Rows returned monitor element" on page 893 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

deadlocks

See monitor element "deadlocks - Deadlocks detected monitor element" on page 614 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

lock_timeouts

See monitor element "lock_timeouts - Number of lock timeouts monitor element" on page 732 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

lock_escals

See monitor element "lock_escals - Number of lock escalations monitor element" on page 721 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_sends_total

See monitor element "fcm_sends_total - FCM sends total monitor element" on page 654 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_recvs_total

See monitor element "fcm_recvs_total - FCM receives total monitor element" on page 650 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_send_volume

See monitor element "fcm_send_volume - FCM send volume monitor element" on page 651 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_recv_volume

See monitor element "fcm_recv_volume - FCM received volume monitor element" on page 647 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_message_sends_total

See monitor element "fcm_message_sends_total - Total FCM message sends monitor element" on page 646 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_message_recvs_total

See monitor element "fcm_message_recvs_total - Total FCM message receives monitor element" on page 643 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_message_send_volume

See monitor element "fcm_message_send_volume - FCM message send volume monitor element" on page 644 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_message_recv_volume

See monitor element "fcm_message_recv_volume - FCM message received volume monitor element" on page 640 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_tq_sends_total

See monitor element "fcm_tq_sends_total - FCM table queue send total monitor element" on page 660 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_tq_recvs_total

See monitor element "fcm_tq_recvs_total - FCM table queue receives total monitor element" on page 657 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_tq_send_volume

See monitor element "fcm_tq_send_volume - FCM table queue send volume monitor element" on page 658 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

fcm_tq_recv_volume

See monitor element "fcm_tq_recv_volume - FCM table queue received volume monitor element" on page 655 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

tq_tot_send_spills

See monitor element "tq_tot_send_spills - Total number of table queue buffers overflowed monitor element" on page 1044 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

post_threshold_sorts

See monitor element "post_threshold_sorts - Post threshold sorts monitor element" on page 858 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

post_shrthreshold_sorts

See monitor element "post_shrthreshold_sorts - Post shared threshold sorts monitor element" on page 856 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

sort_overflows

See monitor element "sort_overflows - Sort overflows monitor element" on page 915 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

audit_events_total

See monitor element "audit_events_total - Total audit events monitor element" on page 531 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_sorts

See monitor element "total_sorts - Total sorts monitor element" on page 1036 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

stmt_exec_time

See monitor element "stmt_exec_time - Statement execution time monitor element" on page 931 for more details.

Contained by: activity_metrics

Element content:

Type	Facet
xs:long	

coord_stmt_exec_time

See monitor element "coord_stmt_exec_time - Execution time for statement by coordinator agent monitor element" on page 591 for more details.

Contained by: activity_metrics

Element content:

Type	Facet
xs:long	

total_routine_non_sect_proc_time

See monitor element "total_routine_non_sect_proc_time - Non-section processing time monitor element" on page 1017 for more details.

Contained by: activity_metrics

Element content:

Type	Facet
xs:long	

total_routine_non_sect_time

See monitor element "total_routine_non_sect_time - Non-section routine execution time monitor elements" on page 1017 for more details.

Contained by: activity_metrics

Element content:

Type	Facet
xs:long	

total_section_proc_time

See monitor element "total_section_proc_time - Total section processing time monitor element" on page 1027 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_section_time

See monitor element "total_section_time - Total section time monitor element" on page 1033 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_app_section_executions

See monitor element "total_app_section_executions - Total application section executions monitor element" on page 993 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_routine_user_code_proc_time

See monitor element "total_routine_user_code_proc_time - Total routine user code processing time monitor element" on page 1020 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_routine_user_code_time

See monitor element "total_routine_user_code_time - Total routine user code time monitor element" on page 1021 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_routine_time

See monitor element "total_routine_time - Total routine time monitor element" on page 1018 for more details.

Contained by: system_metricsactivity_metrics

Element content:

Type	Facet
xs:long	

thresh_violations

See monitor element "thresh_violations - Number of threshold violations monitor element" on page 981 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

num_lw_thresh_exceeded

See monitor element "num_lw_thresh_exceeded - Number of lock wait thresholds exceeded monitor element" on page 778 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

total_routine_invocations

See monitor element "total_routine_invocations - Total routine invocations monitor elements" on page 1016 for more details.

Contained by: system_metrics activity_metrics

Element content:

Type	Facet
xs:long	

Event monitors that write to tables, files, and pipes

Some event monitors can be configured to write information about database events to tables, pipes, or files.

Event monitors are used to collect information about the database and any connected applications when specified events occur. Events represent transitions in database activity such as connections, deadlocks, statements, or transactions. You can define an event monitor by the type of event or events you want it to monitor. For example, a deadlock event monitor waits for a deadlock to occur; when one does, it collects information about the applications involved and the locks in contention.

To create an event monitor, use the CREATE EVENT MONITOR SQL statement. Event monitors collect event data only when they are active. To activate or deactivate an event monitor, use the SET EVENT MONITOR STATE SQL

statement. The status of an event monitor (whether it is active or inactive) can be determined by the SQL function EVENT_MON_STATE.

When the CREATE EVENT MONITOR SQL statement is executed, the definition of the event monitor it creates is stored in the following database system catalog tables:

- SYSCAT.EVENTMONITORS: event monitors defined for the database.
- SYSCAT.EVENTS: events monitored for the database.
- SYSCAT.EVENTTABLES: target tables for table event monitors.

Each event monitor has its own private logical view of the instance's data in the monitor elements. If a particular event monitor is deactivated and then reactivated, its view of these counters is reset. Only the newly activated event monitor is affected; all other event monitors will continue to use their view of the counter values (plus any new additions).

Event monitor output can be directed to non-partitioned SQL tables, a file, or a named pipe.

Note: The deprecated detailed deadlock event monitor, DB2DETAILDEADLOCK, is created by default for each database and starts when the database is activated. Avoid the overhead this event monitor incurs by dropping it. The use of the DB2DETAILDEADLOCK monitor element is no longer recommended. This deprecated event monitor might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Collecting information about database system events

Event monitors are used to collect information about the database and any connected applications when specified events occur. Event monitors are database objects, and as such, they are created and manipulated using SQL data definition language (SQL DDL) statements.

Before you begin

You will need SQLADM or DBADM authority to create and manipulate event monitors.

About this task

A certain type of event monitor gives you the choice to direct the output to a file, pipe, or (regular) table. This type of monitor is described here. Some of the these details do not apply to event monitors that direct output to an unformatted event table.

The steps listed below represent a typical life cycle of an event monitor. These steps need not necessarily be executed in the presented order, if at all. For instance, depending on usage it is possible that an event monitor is never dropped or even deactivated. There are, however, two constants in an event monitor's life cycle: the first step will always be the creation of the event monitor and the last step will always be the deletion of the event monitor.

Procedure

1. "Creating an event monitor" on page 200.
2. *For file and pipe event monitors only:*
   - Ensure that the directory or named pipe that will receive the event records exists. The event monitor will not activate otherwise.

     On AIX®, you can create named pipes by using the mkfifo command. On Linux and other UNIX types (such as the Solaris operating system) use the pipe() routine.

     On Windows, you can create named pipes by using the CreateNamedPipe() routine.
   - *For pipe event monitors only*: Open the named pipe prior to activating the event monitor. This can be done with an operating system function:
     – for UNIX: `open()`
     – for Windows: `ConnectNamedPipe()`

     This can also be done with the db2evmon executable:

     ```
     db2evmon -db databasename
              -evm eventmonname
     ```

     `databasename` represents the name of the database being monitored.

     `evmonname` represents the name of the event monitor.
3. Activate the newly created event monitor to enable it to collect information.

   ```
   SET EVENT MONITOR evmonname STATE 1;
   ```

   If the event monitor was created with the AUTOSTART option, the event monitor will be activated when the first user connects to the database. Furthermore, once an event monitor has been explicitly activated, it will automatically restart whenever the database is reactivated. Restarting occurs until the event monitor is explicitly deactivated or until the instance is stopped. When a table event monitor is started, the event monitor updates the evmon_activates column of the SYSCAT.EVENTMONITORS catalog table. This change is logged, so the DATABASE CONFIGURATION will display:

   ```
   All committed transactions have been written to disk = NO
   ```

   If an event monitor is created with the AUTOSTART option, and the first user connects to the database and immediately disconnects so that the database is deactivated, a log file will be produced.
4. To see if an event monitor is active or inactive, issue the SQL function EVENT_MON_STATE in a query against the table, SYSCAT.EVENTMONITORS:

   ```
   SELECT evmonname, EVENT_MON_STATE(evmonname) FROM
          syscat.eventmonitors;
   ```

   A list of all existing event monitors will be listed, along with their status. A returned value of 0 indicates that the specified event monitor is inactive, and 1 indicates that it is active.
5. Read event monitor output. For write-to-to table event monitors this involves examining the target tables. To access file or pipe event monitor data from the CLP see the Related task, Formatting file or pipe event monitor output from a command line".
6. To deactivate, or turn off an event monitor, use the SET EVENT MONITOR statement:

   ```
   SET EVENT MONITOR evmonname STATE 0
   ```

Deactivating an event monitor does not result in its deletion. It will exist as a dormant database object. Deactivating an event monitor will flush all its contents. Hence, if you reactivate a deactivated event monitor, it will only contain information collected since its reactivation.

Note that if an activity event monitor is active when the database deactivates, any backlogged activity records in the queue are discarded. To ensure that you obtain all activity event monitor records and that none are discarded, explicitly deactivate the activity event monitor first before deactivating the database. When an activity event monitor is explicitly deactivated, all backlogged activity records in the queue are processed before the event monitor deactivates.

7. After you deactivate a pipe event monitor, close the corresponding named pipe. In UNIX use the `close()` function, and in Windows 2000 use the `DisconnectNamedPipe()` function.
8. To eliminate an event monitor object, use the DROP EVENT MONITOR statement:

   ```
   DROP EVENT MONITOR evmonname
   ```

   You can only drop an event monitor if it is inactive.
9. After you drop a pipe event monitor, delete the corresponding named pipe. In UNIX use the `unlink()` function, and in Windows use the `CloseHandle()` function. When dropping a write-to-table event monitor, the associated target tables are not dropped. Similarly, when dropping a file event monitor, the associated files are not deleted.

Creating an event monitor

The first step in an event monitor's life cycle is its creation. Before you create an event monitor, you must determine where the event records are to be sent: to SQL tables, files, or through named pipes.

Before you begin

You will need SQLADM or DBADM authority to create an event monitor.

About this task

For each event record destination there are particular options that are to be specified in the CREATE EVENT MONITOR SQL statement. The target table of a CREATE EVENT MONITOR statement must be a non-partitioned table. Monitoring events in a partitioned database also requires special attention.

A certain type of event monitor gives you the choice to direct the output to a file, pipe, or (regular) table. This type of monitor is described here. Some of the these details do not apply to event monitors that direct output to an unformatted event table.

Procedure

1. Create a table event monitor.
2. Create a file event monitor.
3. Create a pipe event monitor.
4. Create an event monitor for a partitioned database.

Results

Once an event monitor is created and activated, it will record monitoring data as its specified events occur.

Creating a table event monitor

When creating an event monitor you must determine where the information it collects is to be stored. A table event monitor streams event records to SQL tables, presenting a simple alternative to file and pipe event monitors in enabling you to easily capture, parse, and manage event monitoring data. For every event type an event monitor collects, target tables are created for each of the associated logical data groups.

Before you begin

You will need SQLADM or DBADM authority to create a table event monitor.

The target table of a CREATE EVENT MONITOR statement must be a non-partitioned table.

About this task

The various options for table event monitors are set in the CREATE EVENT MONITOR statement. For further assistance in generating CREATE EVENT MONITOR SQL statements for write-to-table event monitors, you can use the **db2evtbl** command. Simply provide the name of the event monitor and the desired event type (or types), and the CREATE EVENT MONITOR statement is generated, complete with listings of all the target tables. You can then copy the generated statement, make modifications, and then execute the statement from the CLP.

Procedure

1. Indicate that event monitor data is to be collected in a table (or set of tables).

   ```
   CREATE EVENT MONITOR dlmon FOR eventtype
                       WRITE TO TABLE
   ```

   `dlmon` is the name of the event monitor.
2. Specify the types of events to be monitored. You can monitor one or more event types with a single event monitor.

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO TABLE
   ```

   This event monitor will monitor for the CONNECTIONS and DEADLOCKS WITH DETAILS event types. Assuming that the above statement was issued by the user 'riihi', the derived names and table spaces of the target tables are as follows:
   - riihi.connheader_dlmon
   - riihi.conn_dlmon
   - riihi.connmemuse_dlmon
   - riihi.deadlock_dlmon
   - riihi.dlconn_dlmon
   - riihi.dllock_dlmon
   - riihi.control_dlmon
3. Specify the size of the table event monitor buffers (in 4K pages) by adjusting the BUFFERSIZE value:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO TABLE BUFFERSIZE 8
```

8 is the combined capacity (in 4K pages) of the two event table buffers. This adds up to 32K of buffer space; 16K for each buffer.

The default size of each buffer is 4 pages (two 16K buffers are allocated). The minimum size is 1 page. The maximum size of the buffers is limited by the size of the monitor heap, because the buffers are allocated from that heap. For performance reasons, highly active event monitors should have larger buffers than relatively inactive event monitors.

4. Indicate if you need the event monitor to be blocked or non-blocked. For blocked event monitors, each agent that generates an event will wait for the event buffers to be written to table if they are full. This can degrade database performance, as the suspended agent and any dependent agents cannot run until the buffers are clear. Use the BLOCKED clause to ensure no losses of event data:

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO TABLE BUFFERSIZE 8 BLOCKED
   ```

   Event monitors are blocked by default.

   If database performance is of greater importance than collecting every single event record, use non-blocked event monitors. In this case, each agent that generates an event will not wait for the event buffers to be written to table if they are full. As a result, non-blocked event monitors are subject to data loss on highly active systems. Use the NONBLOCKED clause to minimize the performance overhead from event monitoring:

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO TABLE BUFFERSIZE 8 NONBLOCKED
   ```

   Note: See "Target tables, control tables and event monitor table management" on page 203 and "Write-to-table and file event monitor buffering" on page 212 for additional information on how information about discarded events is written to the control table for the event monitor.

5. Indicate the logical data groups from which you need to collect event records. Event monitors store the data from each logical data group in corresponding tables.

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO TABLE CONN, DLCONN, DLLOCK
                       BUFFERSIZE 8 NONBLOCKED
   ```

   The CONN, DLCONN, and DLLOCK logical data groups are selected. Not mentioned are the other available logical data groups, CONNHEADER, DEADLOCK, or CONTROL. Data relevant to CONNHEADER, DEADLOCK, or CONTROL will not be stored for the `dlmon` event monitor.

6. Indicate the monitor elements for which you need to collect data.

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                WRITE TO TABLE CONN,
                DLCONN (EXCLUDES(agent_id, lock_wait_start_time)),
                DLLOCK (INCLUDES(lock_mode, table_name))
                BUFFERSIZE 8 NONBLOCKED
   ```

   All the monitor elements for CONN are captured (this is the default behavior). For DLCONN, all monitor elements except **agent_id** and **lock_wait_start_time** are captured. And finally, for DLLOCK, **lock_mode** and **table_name** are the only monitor elements captured.

7. Provide names for the tables to be created, and designate a table space:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
WRITE TO TABLE CONN,
DLCONN (TABLE mydept.dlconnections
EXCLUDES(agent_id, lock_wait_start_time)),
DLLOCK (TABLE dllocks IN mytablespace
INCLUDES(lock_mode, table_name))
BUFFERSIZE 8 NONBLOCKED
```

   Assuming that the above statement was issued by the user `riihi`, the derived names and table spaces of the target tables are as follows:

   - CONN: riihi.conn_dlmon (on the default table space)
   - DLCONN: mydept.dlconnections (on the default table space)
   - DLLOCK: riihi.dllocks (on the MYTABLESPACE table space)

   The default table space is assigned from IBMDEFAULTGROUP, provided the event monitor definer has USE privileges. If the definer does not have USE privileges over this table space, then a table space over which the definer does have USE privileges will be assigned.

8. Indicate how full the table space can get before the event monitor automatically deactivates:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO TABLE DLCONN PCTDEACTIVATE 90
                    BUFFERSIZE 8 NONBLOCKED
```

   When the table space reaches 90% capacity, the dlmon event monitor automatically shuts off. The PCTDEACTIVATE clause can only be used for DMS table spaces. If the target table space has auto-resize enabled, set the PCTDEACTIVATE clause to 100.

9. Specify if the event monitor is to be activated automatically each time the database starts. By default, event monitors (with the exception of the WLM event monitors) are not activated automatically when the database starts.

   - To create an event monitor that starts automatically when the database starts, issue the following statement:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO TABLE BUFFERSIZE 8 NONBLOCKED
                    AUTOSTART NONBLOCKED
```

   - To create an event monitor that does not start automatically when the database starts, issue the following statement:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO TABLE BUFFERSIZE 8 NONBLOCKED
                    MANUALSTART
```

10. To activate or deactivate an event monitor, use the SET EVENT MONITOR STATE statement.

Results

Once a table event monitor is created and activated, it will record monitoring data as its specified events occur.

Target tables, control tables and event monitor table management

You can define an event monitor to store its event records in SQL tables. To do this, use the CREATE EVENT MONITOR statement with the WRITE TO TABLE clause.

Upon the creation of a write-to-table event monitor, the database creates *target tables* to store records for each of the logical data groups returning data. By default, the database creates the tables in the event monitor creator's schema, and names the tables according to their corresponding logical data group and event monitor name. In each table, the column names match the monitor element names that they represent.

For example, the user `riihi` is creating an event monitor that captures STATEMENTS events:

```
CREATE EVENT MONITOR foo FOR STATEMENTS WRITE TO TABLE
```

Event monitors using the STATEMENTS event type collect data from the event_connheader, event_stmt, and event_subsection logical data groups. In addition to the tables representing logical data groups specific to individual event types, a control table is created for every write-to-table event monitor. So, for the event monitor foo, created by user `riihi`, the database creates the following tables:

- riihi.connheader_foo
- riihi.stmt_foo
- riihi.subsection_foo
- riihi.control_foo

The first three tables correspond to each of the logical data groups event_connheader, event_stmt, and event_subsection. The last table, riihi.control_foo is the control table. The control table contains event monitor metadata, specifically, from the event_start, event_db_header (conn_time monitor element only), and event_overflow logical data groups.

Monitor elements are written to the event_overflow group only for *non-blocked* event monitors. In non-blocked event monitors, agents that generate events do not wait for the event buffers to be written to the table if the buffers are full. Instead, they discard monitor data coming from agents when data is coming faster than the event monitor can write the data. In this case, the event monitor records information in the control table to indicate that an overflow has taken place. Included in this information is the monitor element **`message`**, which in the event of an overflow, contains the text `OVERFLOW:n`, where *n* represents the number of event records that were discarded because the event buffers were full.

Whenever a write-to-table event monitor activates, it will acquire IN table locks on each target table in order to prevent them from being modified while the event monitor is active. Table locks are maintained on all tables while the event monitor is active. If exclusive access is required on any of the target tables (for example, when a utility is to be run), first deactivate the event monitor to release the table locks before attempting such access.

Each column name in a target table matches an event monitor element identifier. Any event monitor element that does not have a corresponding target table column is ignored.

Write-to-table event monitor target tables, including the unformatted event (UE) table, must be pruned manually. On highly active systems, event monitors can quickly fill machine space due to the high volume of data they record. Unlike event monitors that write to files or named pipes, you can define write-to-table event monitors to record only certain logical data groups, or monitor elements. This feature enables you to collect only the data relevant to your purposes and reduce the volume of data generated by the event monitors. For example, the

following statement defines an event monitor that captures TRANSACTIONS events, but only from the event_xact logical data group, and including only the **lock_escal** monitor element:

```
CREATE EVENT MONITOR foo_lite FOR TRANSACTIONS WRITE TO TABLE
                    XACT(INCLUDES(lock_escal))
```

Note: This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR UNIT OF WORK statement to monitor transaction events.

There are circumstances where it may not be desirable to have the event monitor's target tables residing in the default schema, with default table names, in the default table space. For instance, you may want the target tables to exist in their own table space if you are anticipating high volumes of monitoring data.

You can specify the schema, table, and table space names in the CREATE EVENT MONITOR statement. The schema name is provided along with the table name, forming a derived name for the table.

A target table can only be used by a single event monitor. If a target table is found to already be defined for another event monitor, or if it cannot be created for any other reason, the CREATE EVENT MONITOR statement will fail.

The table space name can be added after the table name with the optional IN clause. Unlike the target tables, which DB2 automatically creates, a table space must already exist if it is included in an event monitor definition. If no table space is specified, then a table space over which the definer has USE privileges will be assigned.

In a partitioned database environment, a write-to-table event monitor will only be active on database partitions where the table space containing the event monitor table exists. When the target table space for an active event monitor does not exist on a particular database partition, the event monitor will be deactivated on that database partition, and an error is written to the **db2diag** log file.

For increased performance in retrieving event monitor data, you can create indexes for the event tables. You can also add additional table attributes, such as triggers, relational integrity, and constraints. The event monitor will ignore them.

For example, the following statement defines an event monitor that captures STATEMENTS events, using the event_connheader, event_stmt, and event_subsection logical data groups. Each of the three target tables has different schema, table and table space combinations:

```
CREATE EVENT MONITOR foo FOR STATEMENTS
WRITE TO TABLE CONNHEADER,
STMT (TABLE mydept.statements),
SUBSECTION (TABLE subsections, IN mytablespace)
```

Assuming that the above statement was issued by the user 'riihi', the derived names and table spaces of the target tables are as follows:

- CONNHEADER: riihi.connheader_foo (on the default table space)
- STMT: mydept.statements (on the default table space)
- SUBSECTION: riihi.subsections (on the MYTABLESPACE table space)

If a target table does not exist when the event monitor activates, activation continues and data that would otherwise be inserted into the target table is ignored. Correspondingly, if a monitor element does not have a dedicated column in the target table, it is ignored.

For active write-to-table event monitors there is a risk that the table spaces storing event records can reach their capacity. To control this risk for DMS table spaces you can define at which percentage of table space capacity the event monitor will deactivate. This can be declared in the PCTDEACTIVATE clause in the CREATE EVENT MONITOR statement.

For SMS table spaces, the value is set to 100. It is recommended that when the target table space has auto-resize enabled the PCTDEACTIVATE be set to 100.

In a non-partitioned database environment, all write to table event monitors are deactivated when the last application terminates (and the database has not been explicitly activated). In a partitioned database environment, write to table event monitors are deactivated when the catalog partition deactivates.

The following table presents the default target table names as sorted by the event type for which they are returned.

Table 14. Write-to-Table Event Monitor Target Tables

Event type	Target table names	Available information
DEADLOCKS[1]	CONNHEADER	Connection metadata
	DEADLOCK	Deadlock data
	DLCONN	Applications and locks involved in deadlock
	CONTROL	Event monitor metadata
DEADLOCKS WITH DETAILS[1]	CONNHEADER	Connection metadata
	DEADLOCK	Deadlock data
	DLCONN	Applications involved in deadlock
	DLLOCK	Locks involved in deadlock
	CONTROL	Event monitor metadata
DEADLOCKS WITH DETAILS HISTORY[1]	CONNHEADER	Connection metadata
	DEADLOCK	Deadlock data
	DLCONN	Applications involved in deadlock
	DLLOCK	Locks involved in deadlock
	STMTHIST	List of the previous statements in the unit of work
	CONTROL	Event monitor metadata
DEADLOCKS WITH DETAILS HISTORY VALUES[1]	CONNHEADER	Connection metadata
	DEADLOCK	Deadlock data
	DLCONN	Applications involved in deadlock
	DLLOCK	Locks involved in deadlock
	STMTHIST	List of the previous statements in the unit of work
	STMTVALS	Input Data values of statements in STMTHIST table
	CONTROL	Event monitor metadata

Table 14. Write-to-Table Event Monitor Target Tables (continued)

Event type	Target table names	Available information
STATEMENT	CONNHEADER	Connection metadata
	STMT	Statement data
	SUBSECTION	Statement data specific to subsection
	CONTROL	Event monitor metadata
TRANSACTIONS[2]	CONNHEADER	Connection metadata
	XACT	Transaction data
	CONTROL	Event monitor metadata
CONNECTIONS	CONNHEADER	Connection metadata
	CONN	Connection data
	CONTROL	Event monitor metadata
	CONNMEMUSE	Memory pool metadata
DATABASE	DB	Database manager data
	CONTROL	Event monitor metadata
	DBMEMUSE	Memory pool metadata
BUFFERPOOLS	BUFFERPOOL	Buffer pool data
	CONTROL	Event monitor metadata
TABLESPACES	TABLESPACE	Tablespace data
	CONTROL	Event monitor metadata
TABLES	TABLE	Table data
	CONTROL	Event monitor metadata
ACTIVITIES	ACTIVITY	Activities that completed executing or were captured in progress.
	ACTIVITYSTMT	Statement information for activities that are statements.
	ACTIVITYVALS	Input data values for activities that have them. The data types that can be reported excludes the following: CLOB, REF, BOOLEAN, STRUCT, DATALINK, LONG VARGRAPHIC, LONG, XMLLOB, and DBCLOB.
	ACTIVITYMETRICS	Activities metrics
	CONTROL	Event monitor metadata
STATISTICS	SCSTATS	Statistics computed from the activities that executed within each service class, work class, or workload in the system.
	WCSTATS	
	WLSTATS	
	HISTOGRAMBIN	
	QSTATS	
	CONTROL	Event monitor metadata
THRESHOLD VIOLATIONS	THRESHOLDVIOLATIONS	List of thresholds that have been violated as well as the times of violations.
	CONTROL	Event monitor metadata

[1] This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

[2] This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR UNIT OF WORK statement to monitor transaction events.

The following logical data groups are not collected for write-to-table event monitors:

- event_log_stream_header
- event_log_header
- event_dbheader (only the **conn_time** monitor element is collected)

The data type of each column in an event monitor table corresponds to the data type of the monitor element represented by the column. The following is a set of data type mappings that correspond the original system monitor data types of the monitor elements (found in `sqlmon.h`) to the SQL data types of the table columns.

Table 15. System Monitor Data Type Mappings

System monitor data type	SQL data type
SQLM_TYPE_STRING	CHAR[n], VARCHAR[n], CLOB[n]
SQLM_TYPE_U8BIT and SQLM_TYPE_8BIT	SMALLINT, INTEGER, or BIGINT
SQLM_TYPE_U16BIT and SQLM_TYPE_16BIT	SMALLINT, INTEGER, or BIGINT
SQLM_TYPE_U32BIT and SQLM_TYPE_32BIT	INTEGER or BIGINT
SQLM_TYPE_U64BIT and SQLM_TYPE_64BIT	BIGINT
SQLM_TIMESTAMP	TIMESTAMP
SQLM_TIME	BIGINT
SQLCA: SQLERRMC	VARCHAR[72]
SQLCA: SQLSTATE	CHAR[5]
SQLCA: SQLWARN	CHAR[11]
SQLCA: other fields	INTEGER or BIGINT
SQLM_TYPE_HANDLE	BLOB[n]

Note:

1. All columns are NOT NULL.
2. Because the performance of tables with CLOB columns is inferior to tables that have VARCHAR columns, consider using the TRUNC keyword when specifying the stmt evmGroup (or dlconn evmGroup, when using deadlocks with details).
3. SQLM_TYPE_HANDLE is used to represent the compilation environment handle object.

Creating a file event monitor

When creating an event monitor you must determine where the information it collects is to be stored. File event monitors store event records in files. File event monitors and their options are defined by the `CREATE EVENT MONITOR` statement.

Before you begin

You will need SQLADM or DBADM authority to create a file event monitor.

About this task

A file event monitor streams event records to a series of 8-character numbered files with the extension "evt" (for example, 00000000.evt, 00000001.evt, and 00000002.evt). The data should be considered to be one logical file even though the data is broken up into smaller pieces (that is, the start of the data stream is the first byte in the file 00000000.evt; the end of the data stream is the last byte in the file nnnnnnnn.evt). An event monitor will never span a single event record across two files.

Procedure

1. Indicate that event monitor data is to be collected in a file (or set of files), and provide a directory location where event files are to be stored.

   ```
   CREATE EVENT MONITOR dlmon FOR eventtype
                       WRITE TO FILE '/tmp/dlevents'
   ```

   `dlmon` is the name of the event monitor.

   `/tmp/dlevents` is the name of the directory path (on UNIX) where the event monitor is to write the event files.

2. Specify the types of events to be monitored. You can monitor one or more event types with a single event monitor.

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO FILE '/tmp/dlevents'
   ```

   This event monitor will monitor for the CONNECTIONS and DEADLOCKS WITH DETAILS event types.

3. Specify the size of the file event monitor buffers (in 4K pages) by adjusting the BUFFERSIZE value:

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
   ```

   8 is the capacity in 4K pages of the two event file buffers.

   The default size of each buffer is 4 pages (two 16K buffers are allocated). The minimum size is 1 page. The maximum size of the buffers is limited by the size of the monitor heap, because the buffers are allocated from that heap. For performance reasons, highly active event monitors should have larger buffers than relatively inactive event monitors.

4. Indicate if you need the event monitor to be blocked or non-blocked. For blocked event monitors, each agent that generates an event will wait for the event buffers to be written to file if they are full. This can degrade database performance, as the suspended agent and any dependent agents cannot run until the buffers are clear. Use the BLOCKED clause to ensure no losses of event data:

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                       BLOCKED
   ```

   Event monitors are blocked by default. If database performance is of greater importance than collecting every single event record, use non-blocked event monitors. In this case, each agent that generates an event will not wait for the event buffers to be written to file if they are full. As a result, non-blocked event monitors are subject to data loss on highly active systems. Use the NONBLOCKED clause to minimize the performance overhead from event monitoring:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                    NONBLOCKED
```

5. Specify the maximum number of event files that can be collected for an event monitor. If this limit is reached, the event monitor will deactivate itself.

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                       NONBLOCKED MAXFILES 5
   ```

   5 is the maximum number of event files that will be created.

   You can also specify that there is no limit to the number of event files that the event monitor can create:

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                       NONBLOCKED MAXFILES NONE
   ```

6. Specify the maximum size (in 4K pages) for each event file created by an event monitor. If this limit is reached, a new file is created.

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                       NONBLOCKED MAXFILES 5 MAXFILESIZE 32
   ```

   32 is the maximum number of 4K pages that an event file can contain.

   This value must be greater than the value specified by the BUFFERSIZE parameter. You can also specify that there is to be no limit on an event file's size:

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                       NONBLOCKED MAXFILES NONE MAXFILESIZE NONE
   ```

7. Specify if the event monitor is to be activated automatically each time the database starts. By default, event monitors (with the exception of the WLM event monitors) are not activated automatically when the database starts.
   - To create an event monitor that starts automatically when the database starts, issue the following statement:

     ```
     CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                         WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                         NONBLOCKED AUTOSTART
     ```

   - To create an event monitor that does not start automatically when the database starts, issue the following statement:

     ```
     CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                         WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                         NONBLOCKED MANUALSTART
     ```

8. To activate or deactivate an event monitor, use the SET EVENT MONITOR STATE statement.

Results

Once a file event monitor is created and activated, it will record monitoring data as its specified events occur.

Event monitor file management

A file event monitor enables the event monitor to store its event records in files. All the output of the event monitor goes in the directory supplied in the FILE parameter for the CREATE EVENT MONITOR statement. Before the monitor is activated, the directory must exist, or the **SET EVENT MONITOR** command returns an

error; the directory is not created by the database manager if it does not exist. When a file event monitor is first activated, a control file named `db2event.ctl` is created in this directory.

Important: Do not remove or modify this file.

By default, an event monitor writes its trace to a single file, called `00000000.evt`. This file keeps growing as long as there is space on the file system. If you specified a file size limit with the **MAXFILESIZE** parameter of the CREATE EVENT MONITOR statement, then when a file is full, output is directed to a new file. The number that makes up the file name is increased by 1 each time a new file is created. Hence, the active file is the file with the highest number.

You can limit the maximum size of the entire event monitor trace by also using the **MAXFILES** parameter of the CREATE EVENT MONITOR statement. When the number of files reaches the maximum defined by MAXFILES, the event monitor deactivates itself and the following message is written to the administration notification log.

```
DIA1601I Event Monitor monitor-name was deactivated when it reached
its preset MAXFILES and MAXFILESIZE limit.
```

If you receive this message, do not delete any of the event monitor files. If you do, you will not be able to view any of the event monitor information (even that contained in any remaining files) using the **db2evmon** command. Instead take one of the following actions:

- Recreate the event monitor without the **MAXFILES** and **MAXFILESIZE** limits.
- Leave the limits imposed by the **MAXFILES** and **MAXFILESIZE** parameters in place, but move all but the most recent *.evt files in the directory to another directory or file system. You can then view the event monitor information from the files in the new directory. You can create a script to do this automatically if you want.

Either way, you must reactivate the event monitor using the statement **SET EVENT MONITOR** *event-monitor-name* **STATE 1** to start collecting information again after you receive the DIA1601I message.

When a file event monitor is restarted, it can either erase any existing data or append new data to it. This option is specified in the CREATE EVENT MONITOR statement, where either an APPEND monitor or a REPLACE monitor can be created. APPEND is the default option. An APPEND event monitor starts writing at the end of the file it was last using. If you have removed that file, the next file number in sequence is used. When an append event monitor is restarted, only a start_event is generated. The event log header and database header are generated only for the first activation. A REPLACE event monitor always deletes existing event files and starts writing at `00000000.evt`.

Note: If you did not use the REPLACE option for the event monitor, you can perform the following steps to force the event monitor to start collecting a new set of data:

1. Deactivate the event monitor using the **SET EVENT MONITOR** *event-monitor-name* **STATE 0** command.
2. Delete all files in the directory that was specified by the FILE option of the CREATE EVENT MONITOR statement.
3. Reactivate the event monitor using the **SET EVENT MONITOR** *event-monitor-name* **STATE 1** command.

If a file event monitor runs out of disk space, it shuts itself down after logging a system-error-level message in the administration notification log.

You might want to process monitor data while the event monitor is active. This is possible, and furthermore, when you are finished processing a file, you can delete it, freeing up space for further monitoring data. An event monitor cannot be forced to switch to the next file unless you stop and restart it. It must also be in APPEND mode. To track which events have been processed in the active file, you can create an application that simply tracks the file number and location of the last record processed. When processing the trace the next time around, the application can seek to that file location.

Write-to-table and file event monitor buffering

For write-to-table and file event monitors, the event monitor process buffers its record before writing them to a file or table. Records are written automatically when a buffer is full. Therefore, you can improve monitoring performance for event monitors with high amounts of throughput by specifying larger buffers to reduce the number of disk accesses. To force an event monitor to flush its buffers, you must either deactivate it or empty the buffers by using the FLUSH EVENT MONITOR statement.

A blocked event monitor suspends the database process that is sending monitor data when both of its buffers are full. This is to ensure that no event records are discarded while the blocked event monitor is active. The suspended database process and consequently, any dependent database processes cannot run until a buffer has been written. This can introduce a significant performance overhead, depending on the type of workload and the speed of the I/O device. Event monitors are blocked by default.

A non-blocked event monitor discards monitor data coming from agents when data is coming faster than the event monitor can write the data. This prevents event monitoring from becoming a performance burden on other database activities.

An event monitor that has discarded event records generates an overflow event. It specifies the start and stop time during which the monitor was discarding events and the number of events that were discarded during that period. It is possible for an event monitor to terminate or be deactivated with a pending overflow to report. If this occurs, the following message is written to the admin log:

```
DIA2503I Event Monitor monitor-name had a pending overflow record
when it was deactivated.
```

Loss of event monitoring data can also occur for individual event records. If the length of an event record exceeds the event buffer size, the data that does not fit in the buffer is truncated. For example, this situation could occur if you are capturing the stmt_text monitor element and applications attached to the database being monitored issue lengthy SQL statements. If you need to capture all of the event record information, specify larger buffers. Keep in mind that larger buffers will result in less frequent writes to file or table.

Creating a pipe event monitor

When creating an event monitor you must determine where the information it collects is to be stored. A pipe event monitor streams event records directly from the event monitor, to a named pipe.

Before you begin

You will need SQLADM or DBADM authority to create a pipe event monitor.

About this task

It is the responsibility of the monitoring application to promptly read the data from the pipe as the event monitor writes the event data. If the event monitor is unable to write data to the pipe (for instance, if it is full), monitor data will be lost.

Pipe event monitors are defined with the CREATE EVENT MONITOR statement.

Procedure

1. Indicate that event monitor data is to be directed to a named pipe.

   ```
   CREATE EVENT MONITOR dlmon FOR eventtype
                       WRITE TO PIPE '/home/riihi/dlevents'
   ```

   dlmon is the name of the event monitor.

   /home/riihi/dlevents is the name of the named pipe (on UNIX) to where the event monitor will direct the event records. The CREATE EVENT MONITOR statement supports UNIX and Windows pipe naming syntax.

   The named pipe specified in the CREATE EVENT MONITOR statement must be present and open when you activate the event monitor. If you specify that the event monitor is to start automatically, the named pipe must exist prior to the event monitor's creation.

2. Specify the types of events to be monitored. You can monitor one or more event types with a single event monitor.

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO PIPE '/home/riihi/dlevents'
   ```

   This event monitor will monitor for the CONNECTIONS and DEADLOCKS WITH DETAILS event types.

3. Specify if the event monitor is to be activated automatically each time the database starts. By default, event monitors are not activated automatically when the database starts.

   - To create an event monitor that starts automatically when the database starts, issue the following statement:

     ```
     CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                         WRITE TO PIPE '/home/riihi/dlevents'
                         AUTOSTART
     ```

   - To create an event monitor that does not start automatically when the database starts, issue the following statement:

     ```
     CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                         WRITE TO PIPE '/home/riihi/dlevents'
                         MANUALSTART
     ```

4. To activate or deactivate an event monitor, use the SET EVENT MONITOR STATE statement.

Results

Once a pipe event monitor is created and activated, it will record monitoring data as its specified events occur.

Event monitor named pipe management

A pipe event monitor enables the processing of the event monitor data stream through a named pipe. Using a pipe event monitor is desirable if you need to process event records in real time. Another important advantage is that your application can ignore unwanted data as it is read off the pipe, giving the opportunity to considerably reduce storage requirements.

On AIX, you can create named pipes by using the mkfifo command. On Linux and other UNIX types (such as the Solaris operating system) use the pipe() routine. On Windows, you can create named pipes by using the CreateNamedPipe() routine.

When you direct data to a pipe, I/O is always blocked and the only buffering is that performed by the pipe. It is the responsibility of the monitoring application to promptly read the data from the pipe as the event monitor writes the event data. If the event monitor is unable to write the data to the pipe (for example, because the pipe is full), monitor data will be lost.

In addition, there must be enough space in the named pipe to handle incoming event records. If the application does not read the data from the named pipe fast enough, the pipe will fill up and overflow. The smaller the pipe buffer, the greater the chance of an overflow.

When a pipe overflow occurs, the monitor creates overflow event records indicating that an overflow has occurred. The event monitor is not turned off, but monitor data is lost. If there are outstanding overflow event records when the monitor is deactivated, a diagnostic message will be logged. Otherwise, the overflow event records will be written to the pipe when possible.

If your operating system allows you to define the size of the pipe buffer, use a pipe buffer of at least 32K. For high-volume event monitors, you should set the monitoring application's process priority equal to or higher than the agent process priority.

Creating an event monitor for partitioned databases

When creating a file or pipe event monitor on partitioned database systems you need to determine the scope of the monitoring data you wish to collect.

Before you begin

You will need SQLADM or DBADM authority to create event monitors for partitioned databases.

About this task

An event monitor uses an operating system process or a thread to write the event records. The database partition where this process or thread runs is called the monitor partition. File and pipe event monitors can be monitoring events as they occur locally on the monitor partition, or globally as they occur on any partition where the DB2 database manager is running. A global event monitor writes a single trace on the monitoring partition that contains activity from all partitions. Whether an event monitor is local or global is referred to as its monitoring scope.

Both the monitor partition and monitor scope are specified with the CREATE EVENT MONITOR statement.

An event monitor can only be activated if the monitor partition is active. If the SET EVENT MONITOR statement is used to activate an event monitor but the monitor partition is not yet active, then event monitor activation will occur when the monitor partition is next started. Furthermore, event monitor activation will automatically occur until the event monitor is explicitly deactivated or the instance is explicitly deactivated. For example, on database partition 0:

```
db2 connect to sample
db2 create event monitor foo ... on dbpartitionnum 2
db2 set event monitor foo state 1
```

After running the above commands, event monitor `foo` will activate automatically whenever the database `sample` activates on database partition 2. This automatic activation occurs until `db2 set event monitor foo state 0` is issued, or partition 2 is stopped.

For write-to-table event monitors, the notion of local or global scope is not applicable. When a write-to-table event monitor is activated, an event monitor process runs on all of the partitions. (More specifically, the event monitor process will run on partitions that belong to the database partition groups in which the target tables reside.) Each partition where the event monitor process is running also has the same set of target tables. The data in these tables will be different as it represents the individual partition's view of the monitoring data. You can get aggregate values from all the partitions by issuing SQL statements that access the desired values in each partition's event monitor target tables.

The first column of each target table is named PARTITION_KEY, and is used as the partitioning key for the table. The value of this column is chosen so that each event monitor process inserts data into the database partition on which the process is running; that is, insert operations are performed locally on the database partition where the event monitor process is running. On any database partition, the PARTITION_KEY field will contain the same value. This means that if a data partition is dropped and data redistribution is performed, all data on the dropped database partition will go to one other database partition instead of being evenly distributed. Therefore, before removing a database partition, consider deleting all table rows on that database partition.

In addition, a column named PARTITION_NUMBER can be defined for each table. This column contains the number of the partition on which the data was inserted. Unlike the PARTITION_KEY column, the PARTITION_NUMBER column is not mandatory.

The table space within which target tables are defined must exist across all partitions that will have event monitor data written to them. Failure to observe this rule will result in records not being written to the log on partitions (with event monitors) where the table space does not exist. Events will still be written on partitions where the table space does exist, and no error will be returned. This behavior allows users to choose a subset of partitions for monitoring, by creating a table space that exists only on certain partitions.

During write-to-table event monitor activation, the CONTROL table rows for FIRST_CONNECT and EVMON_START are inserted on all database partitions where the table space for target tables exists.

If a partition is not yet active when a write-to-table event monitor is activated, the event monitor will be activated when that partition next activates.

If you add a database partition that is online immediately after being added, the event monitors are not immediately aware of the new partition. To collect and record data about the new partition, you must do one of the following:

- For global event monitors, restart the event monitors.
- For write-to-table event monitors, drop, recreate, and restart the event monitors.

.

Note: The lock list in the detailed deadlock connection event will only contain the locks held by the application on the partition where it is waiting for the lock. For example, if an application involved in a deadlock is waiting for a lock on node 20, only the locks held by that application on node 20 will be included in the list.

Procedure

1. Specify the partition to be monitored.

   ```
   CREATE EVENT MONITOR dlmon FOR DEADLOCKS
                       WRITE TO FILE '/tmp/dlevents'
                       ON PARTITION 3
   ```

   `dlmon` represents the name of the event monitor.

   `/tmp/dlevents` is the name of the directory path (on UNIX) where the event monitor is to write the event files.

   `3` represents the partition number to be monitored.

2. Specify if the event monitor data is to be collected at a local or global scope. To collect event monitor reports from all partitions issue the following statement:

   ```
   CREATE EVENT MONITOR dlmon FOR DEADLOCKS
                       WRITE TO FILE '/tmp/dlevents'
                       ON PARTITION 3 GLOBAL
   ```

   Only deadlock and deadlock with details event monitors can be defined as GLOBAL. All partitions will report deadlock-related event records to partition 3.

3. To collect event monitor reports from only the local partition issue the following statement:

   ```
   CREATE EVENT MONITOR dlmon FOR DEADLOCKS
                       WRITE TO FILE '/tmp/dlevents'
                       ON PARTITION 3 LOCAL
   ```

   This is the default behavior for file and pipe event monitors in partitioned databases. The LOCAL and GLOBAL clauses are ignored for write-to-table event monitors.

4. It is possible to review the monitor partition and scope values for existing event monitors. To do this query the SYSCAT.EVENTMONITORS table with the following statement:

   ```
   SELECT EVMONNAME, NODENUM, MONSCOPE FROM SYSCAT.EVENTMONITORS
   ```

Results

Once an event monitor is created and activated, it will record monitoring data as its specified events occur.

Event monitor sample output

Formatting file or pipe event monitor output from a command line

The output of a file or pipe event monitor is a binary stream of logical data groupings. You can format this data stream from a command line by using the db2evmon command. This productivity tool reads in event records from an event monitor's files or pipe, then writes them to the screen (standard output).

Before you begin

No authorization is required unless you are connecting to the database, in which case one of the following is required:

- SYSADM
- SYSCTRL
- SYSMAINT
- DBADM

About this task

You can indicate which event monitor's output you will format by either providing the path of the event files, or providing the name of the database and the event monitor's name. To format event monitor output:

Procedure

- Specify the directory containing the event monitor files:

  ```
  db2evmon -path '/tmp/dlevents'
  ```

  `/tmp/dlevents` represents a (UNIX) path.
- Specify the database and event monitor name:

  ```
  db2evmon -db 'sample' -evm 'dlmon'
  ```

  `sample` represents the database the event monitor belongs to.

  `dlmon` represents an event monitor.

Event records and their corresponding applications

In an event trace for an active database with hundreds of attached applications, it can be tedious to track event records associated with a specific application. For traceability, each event record includes the application handle and application ID. These allow you to correlate each record with the application for which the event record was generated.

The application handle (**agent_id**) is unique system-wide for the duration of the application. However, it will eventually be reused (a 16 bit counter is used to generate this identifier -- on partitioned database systems this consists of the coordinating partition number and a 16 bit counter). In most cases, this reuse is not a problem, since an application reading records from the trace is able to detect a connection that was terminated. For example, encountering (in the trace) a connection header with a known agent_ID implies that the previous connection with this agent_ID was terminated.

The application ID is a string identifier that includes a timestamp and is guaranteed to remain unique, even after stopping and restarting the database manager.

Finding event records for a certain application is particularly easy with write-to-table event monitors. In the event monitor tables, where each row corresponds to an event record, the application handle and application ID are default column values. To find all the event records for a given application, you can simply issue an SQL select statement for all event records corresponding to the particular application ID.

Event monitor self-describing data stream

An event monitor that writes to a pipe or file outputs a binary stream of logical data groupings that are exactly the same for both pipe and file event monitors. You can format the data stream either by using the **db2evmon** command or by developing a client application. This data stream is presented in a self-describing format. Figure 5 shows the structure of the data stream and Table 16 on page 219 provides some examples of the logical data groups and monitor elements that could be returned.

Note: In the examples and tables descriptive names are used for the identifiers. These names are prefixed by **SQLM_ELM_** in the actual data stream. For example, **db_event** would appear as SQLM_ELM_DB_EVENT in the event monitor output. Types are prefixed with **SQLM_TYPE_** in the actual data stream. For example, headers appear as SQLM_TYPE_HEADER in the data stream.

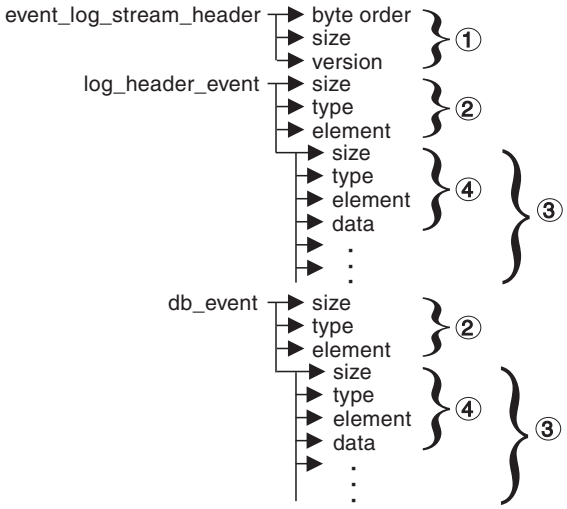


Figure 5. Pipe or File Event Monitor Data Stream

1. The structure of the sqlm_event_log_data_stream_header is different than the other headers in the data stream. The version field determines if the output can be processed as a self-describing data stream.

   This header has the same size and type as pre-Version 6 event monitor streams. This allows applications to determine if the event monitor output is self-describing or is in the pre-Version 6 static format.

   Note: This monitor element is extracted by reading sizeof(sqlm_event_log_data_stream) bytes from the data stream.
2. Each logical data group begins with a header that indicates its size and element name. This does not apply event_log_stream_header, as its size element contains a dummy value to maintain backwards compatibility.
3. The size element in the header indicates the size of all the data in that logical data group.

4. Monitor element information follows its logical data group header and is also self-describing.

Table 16. Sample event data stream

Logical Data Group	Data Stream	Description
event_log_stream_header	┬►sqlm_little_endian	Not used (for compatibility with previous releases).
	├►200	Not used (for compatibility with previous releases).
	└►sqlm_dbmon_version9	The version of the database manager that returned the data. Event monitors write data in the self-describing format.
log_header_event	┬►100	Size of the logical data group.
	├►header	Indicates the start of a logical data group.
	├►log_header	Name of the logical data group.
	├ →►4	Size of the data stored in this monitor element.
	│ ├►u32bit	Monitor element type - 32 bit numeric.
	│ ├►byte_order	The name of the monitor element collected.
	│ └►little_endian	The collected value for this element.
	└ ┬►2	Size of the data stored in this monitor element.
	├►u16bit	Monitor element type - unsigned 16 bit numeric.
	├►codepage_id	The name of the monitor element collected.
	└►850	The collected value for this element.
db_event	┬►100	Size of the logical data group.
	├►header	Indicates the start of a logical data group.
	├►db_event	Name of the logical data group.
	└ →►4	Size of the data stored in this monitor element
	├►u32bit	Monitor element type - unsigned 32 bit numeric.
	├►lock_waits	The name of the monitor element collected.
	└►2	The collected value for this element.

The event_log_stream_header identifies the version of the database manager that returned the data. Event monitors write their data in the self-describing format. An event monitor, unlike a snapshot monitor, does not have a **size** element that returns the total size of the trace. The number present in event_log_stream_header is a dummy value present for backwards compatibility. The total size of an event trace is not known when the event_log_stream_header is written. You typically read an event monitor trace until you reach an end of file or pipe.

The log header describes the characteristics of the trace, containing information such as the memory model (for example little endian) of the server where the trace was collected, and the code page of the database. You might have to do byte swapping on numerical values, if the system where you read the trace has a different memory model than the server (for example, if you are reading a trace from a UNIX server on a Windows 2000 system). Code page translation might also need to be done if the database is configured in a different language than the machine from which you read the trace. When reading the trace, you can use the **size** element to skip a logical data group in the trace.

Transferring event monitor data between systems

When transferring event monitor information between systems using different conventions for storing numerical values, conversions must be made. Information on UNIX platforms is stored in little endian byte order, and information on Windows platforms is stored in big endian byte order. If event monitor data from a little endian source is to be read on a big endian platform, or vice versa, byte conversion is necessary.

Procedure

1. To convert the numeric values in logical data group headers and monitor elements use the following logic (presented in C):

```
#include sqlmon.h
#define SWAP2(s) ((((s) >> 8) & 0xFF) | (((s) << 8) & 0xFF00))

#define SWAP4(l) ((((l) >> 24) & 0xFF) | ((((l) & 0xFF0000) >> 8) & 0xFF00) \
                     | (((l) & 0xFF00) << 8) | (((l) << 24))

#define SWAP8( where )                                                    \
 {                                                                        \
    sqluint32 temp;                                                       \
    temp = SWAP4(*(sqluint32 *) (where));                                 \
    * (sqluint32 *) (where) = SWAP4(* (((sqluint32 *) (where)) + 1));     \
    * (((sqluint32 *) (where)) + 1) = temp;                               \
 }

int HeaderByteReverse( sqlm_header_info * pHeader)
{   int rc = 0;

    pHeader->size = SWAP4(pHeader->size);
    pHeader->type = SWAP2(pHeader->type);
    pHeader->element = SWAP2(pHeader->element);

    return rc;
}

int DataByteReverse( char * dataBuf, sqluint32 dataSize)
{   int rc = 0;

    sqlm_header_info * pElemHeader = NULL;
    char * pElemData = NULL;
    sqluint32 dataOffset = 0;
    sqluint32 elemDataSize  = 0;
    sqluint32 elemHeaderSize = sizeof( sqlm_header_info);

    // For each of the elements in the data stream that are numeric,
    // perform byte reversal.

    while( dataOffset < dataSize)
    {   /* byte reverse the element header */
        pElemHeader = (sqlm_header_info *)
                      ( dataBuf + dataOffset);

        rc = HeaderByteReverse( pElemHeader);
        if( rc != 0) return rc;
        // Remember the element data's size...it will be byte reversed
        // before we skip to the next element.
        elemDataSize = pElemHeader->size;

        /* byte reverse the element data */
        pElemData = (char *)
                      ( dataBuf + dataOffset + elemHeaderSize);

        if(pElemHeader->type == SQLM_TYPE_HEADER)
        {   rc = DataByteReverse( pElemData, pElemHeader->size);
            if( rc != 0) return rc;
        }
        else
 {   switch( pElemHeader->type)
           {    case SQLM_TYPE_16BIT:
                case SQLM_TYPE_U16BIT:
                    *(sqluint16 *)(pElemData) =
                                    SWAP2(*(short *)(pElemData));
                    break;
                case SQLM_TYPE_32BIT:
```

```
                case SQLM_TYPE_U32BIT:
                    *(sqluint32 *)(pElemData) =
                                    SWAP4(*(sqluint32 *)(pElemData));
                    break;
                case SQLM_TYPE_64BIT:
                case SQLM_TYPE_U64BIT:
                    SWAP8(pElemData);
                    break;
                default:
                    // Not a numeric type. Do nothing.
                    break;
            }
        }
        dataOffset = dataOffset + elemHeaderSize + elemDataSize;
    }

    return 0;
} /*  end of DataByteReverse */
```

2. To convert the numeric values in logical data group headers and monitor elements use the following logic (presented in C):

```
#include sqlmon.h
#define SWAP2(s) ((((s) >> 8) & 0xFF) | (((s) << 8) & 0xFF00))

#define SWAP4(l) ((((l) >> 24) & 0xFF) | ((((l) & 0xFF0000) >> 8) & 0xFF00) \
                     | (((l) & 0xFF00) << 8) | (((l) << 24))

#define SWAP8( where )                                                      \
 {                                                                          \
    sqluint32 temp;                                                         \
    temp = SWAP4(*(sqluint32 *) (where));                                   \
    * (sqluint32 *) (where) = SWAP4(* (((sqluint32 *) (where)) + 1));       \
    * (((sqluint32 *) (where)) + 1) = temp;                                 \
 }

int HeaderByteReverse( sqlm_header_info * pHeader)
{   int rc = 0;

    pHeader->size = SWAP4(pHeader->size);
    pHeader->type = SWAP2(pHeader->type);
    pHeader->element = SWAP2(pHeader->element);

    return rc;
}

int DataByteReverse( char * dataBuf, sqluint32 dataSize)
{   int rc = 0;

    sqlm_header_info * pElemHeader = NULL;
    char * pElemData = NULL;
    sqluint32 dataOffset = 0;
    sqluint32 elemDataSize  = 0;
    sqluint32 elemHeaderSize = sizeof( sqlm_header_info);

    // For each of the elements in the data stream that are numeric,
    // perform byte reversal.

    while( dataOffset < dataSize)
    {   /* byte reverse the element header */
        pElemHeader = (sqlm_header_info *)
                      ( dataBuf + dataOffset);

        rc = HeaderByteReverse( pElemHeader);
        if( rc != 0) return rc;
        // Remember the element data's size...it will be byte reversed
        // before we skip to the next element.
        elemDataSize = pElemHeader->size;
```

```
        /* byte reverse the element data */
        pElemData = (char *)
                      ( dataBuf + dataOffset + elemHeaderSize);

        if(pElemHeader->type == SQLM_TYPE_HEADER)
        {   rc = DataByteReverse( pElemData, pElemHeader->size);
            if( rc != 0) return rc;
        }
        else
 {   switch( pElemHeader->type)
           {    case SQLM_TYPE_16BIT:
                case SQLM_TYPE_U16BIT:
                    *(sqluint16 *)(pElemData) =
                                      SWAP2(*(short *)(pElemData));
                    break;
                case SQLM_TYPE_32BIT:
                case SQLM_TYPE_U32BIT:
                    *(sqluint32 *)(pElemData) =
                                      SWAP4(*(sqluint32 *)(pElemData));
                    break;
                case SQLM_TYPE_64BIT:
                case SQLM_TYPE_U64BIT:
                    SWAP8(pElemData);
                    break;
                default:
                    // Not a numeric type. Do nothing.
                    break;
           }
        }
        dataOffset = dataOffset + elemHeaderSize + elemDataSize;
    }

    return 0;
} /*  end of DataByteReverse */
```

Chapter 5. Reports generated using the MONREPORT module

The MONREPORT module generates text reports of monitoring data that you can use to troubleshoot SQL performance problems.

You can generate the following reports using the MONREPORT module:

Table 17. List of reports generated using the MONREPORT module

Report Name	Procedure to create report	Main data source / table functions
Summary report	MONREPORT.DBSUMMARY	MON_GET_SERVICE_SUBCLASS and selected details from MON_GET_CONNECTION and MON_GET_WORKLOAD
Connection report	MONREPORT.CONNECTION	MON_GET_CONNECTION
Current Applications report	MONREPORT.CURRENTAPPS	Includes fields from MON_GET_CONNECTION, MON_GET_UNIT_OF_WORK, WLM_GET_SERVICE_CLASS_AGENTS_V97, WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES_V97
Current SQL report	MONREPORT.CURRENTSQL	WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES_V97 and statement text obtained from MON_GET_PKG_CACHE_STMT for the specified **executable_id** values
Package Cache report	MONREPORT.PKGCACHE	MON_GET_PKG_CACHE_STMT
Current Lock Wait report	MONREPORT.LOCKWAIT	Most data from MON_GET_APPL_LOCKWAIT; additional data from MON_GET_CONNECTION, WLM_GET_SERVICE_CLASS_AGENTS_V97, WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES_V97, MON_GET_PKG_CACHE_STMT, MON_GET_TABLE

To generate a report, issue a command such as `call monreport.connection(60, 32)`. The first parameter is the monitoring interval; the subsequent parameters are report-specific. Use the same monitoring interval for all reports to establish accurate baseline values and consistent results.

Most reports start with a summary section that provides one line of key information for each item in the report. For example, the Connection report contains a one-line summary of each connection. The main body of the report consists of a detailed section for each item in the summary.

Each metric in the report is labeled with the underlying monitor element name (for example: CLIENT_IDLE_WAIT_TIME = 44). To determine what the metric represents, search the Information Center for the monitor element name.

You can customize the reports generated by the MONREPORT module. The MONREPORT module is implemented entirely using SQL and you can obtain the module code from the database catalog and create a customized version.

© Copyright IBM Corp. 1993, 2012

Reports for initial diagnosis

An important use of these reports is to troubleshoot SQL performance slowdowns. Each report is designed to answer certain diagnosis questions. Some reports support initial diagnosis, while others support subsequent detailed diagnosis of particular types of problems.

Initial diagnosis involves:

- Determining the problem category, by narrowing the problem down to the aspect or stage of processing that has slowed down.
- Identifying the SQL statements involved in the problem and collecting information about the SQL statements for further analysis.

Table 18. MONREPORT module reports suitable for initial diagnosis

Procedure name	Information provided and usage
MONREPORT.DBSUMMARY Part 1: System Performance	Part 1 of the Summary report provides monitor data for most aspects of processing aggregated across the entire database. This information is useful for answering questions about the aspect or stage of processing that has slowed down. For example: • Is the problem inside or outside the data server? • Is there a computing resource bottleneck? • Are requests in a wait state? If so, for what resource? • Is the slowdown located in a particular data server processing component?
MONREPORT.DBSUMMARY Part 2: Application Performance	Part 2 of the Summary report provides key performance indicators for each connection, workload, and service class. This information is useful for answering questions about the scope of application requests involved in the slowdown. For example: • Is this slowdown a general system slowdown that affects much or all the workload? • Is this slowdown limited to SQL statements issued from a particular source such as particular connections, DB2 workloads or DB2 service classes?
MONREPORT.DBSUMMARY Part 3: Member level information	Part 3 of the Summary report provides key performance indicators for each member. This information is useful for determining whether the slowdown is isolated to one or a few members.

Table 18. MONREPORT module reports suitable for initial diagnosis (continued)

Procedure name	Information provided and usage
MONREPORT.CURRENTSQL	The Current SQL report provides information about statements that are currently running, in the form of several lists of the top *N* activities. Each list ranks activities by a different metric, such as processing resources, rows processed, number of direct reads and direct writes. This information is useful for determining whether the slowdown is isolated to one or a few SQL statements. If the slowdown is isolated to one or a few SQL statements, those statements are likely to appear in this report of top statements.
MONREPORT.PKGCACHE	The Package Cache report provides information about statements that have run recently and are stored in the package cache. This report shows several summaries, each listing the top *N* activities. The activities are ranked by the following metrics: • CPU • wait time • rows processed • number of executions • I/O wait time For each of CPU, wait time, rows processed, and I/O wait time, the report shows two summaries: one for the metric aggregated across all executions of a statement, and one for the metric averaged by the number of executions of the statement. This information is useful for determining whether the slowdown is isolated to one or a few SQL statements. If so, those statements are likely to appear at the top in this report. The information per execution can help identify the most costly statements while the information summed across executions can help identify statements with the most impact on the system cumulatively considering both the statement cost and frequency of execution.

Table 18. MONREPORT module reports suitable for initial diagnosis (continued)

Procedure name	Information provided and usage
MONREPORT.CURRENTAPPS	The Current Applications report shows the current processing state for units of work, agents, and activities. The report starts with a summary section showing the number of current connections and activities, as well as a series of summaries, such as the summary of current units of work by workload occurrence state. The body of the report consists of one section for each connection that provides the details of the connection. This information is useful for viewing all the work currently running on the system. This allows you to check for patterns that might identify the problem category.

Reports for detailed diagnosis

After completing the initial diagnosis, you might need to pursue a specialized or detailed set of troubleshooting analyses for the problem category you identified during the initial diagnosis phase.

Table 19. MONREPORT module reports suitable for detailed diagnosis

Procedure name	Information provided and usage
MONREPORT.CONNECTION	If the Summary report showed that the slowdown is limited to SQL statements issued from a particular connection, then you can view detailed information about the affected connection. This report contains the same metrics as Part 1 of the Summary report, but it presents this information for each connection.
MONREPORT.LOCKWAIT	If the reports viewed during the initial diagnosis suggest there is a lock wait problem, then you can view detailed information about each lock wait currently in progress. This information includes lock holder and lock requestor details, as well as characteristics of the lock held and the lock requested.

Diagnosing SQL performance slowdowns using MONREPORT module reports

To effectively diagnose SQL performance slowdowns, use the Summary report to determine the problem category, and then use other reports generated by the MONREPORT module to gather additional information about the slowdown.

Before you begin

To diagnose performance problems using the MONREPORT module reports, it is helpful to compare current monitor element values to typical values. Typical values can be determined from best practices, baseline values collected previously, and current values on a comparable system. If you regularly generate reports when your system is not experiencing performance problems, then you will have a ready set of baseline data which you can compare to the reports generated during troubleshooting.

Perform any or all of the following steps.

Procedure

- Review system performance monitor data to determine the problem category.
- Identify the scope of the slowdown. Determine whether the slowdown affects all or most of the workload; whether it is limited to SQL statements issued from a particular connection, workload, or service class; or whether it is limited to a particular database member.
- Review the top SQL statements and determine whether they are involved in the slowdown. You can also obtain execution history and other information about the SQL statements.
- Review all work running on the system. Check for patterns that might determine the problem category and identify the SQL statements involved in the slowdown.
- Collect statement text and details of the SQL statements involved.

Determining the problem category

When troubleshooting SQL performance slowdowns, you can use monitoring information to narrow down the problem to a specific location or phase of processing.

About this task

Use the monitoring information in the Summary report to find out the pattern of symptoms for the SQL slowdown incident. With the monitoring information, you can determine which category the problem belongs to, and use this information to either resolve the problem or continue the diagnosis.

Procedure

1. Generate the Summary report. Issue the **call monreport.dbsummary** command. For example, to generate the Summary report that collects system and application performance metrics and then collects them again in 45 seconds, issue the following command:

   ```
   call monreport.dbsummary(45);
   ```

2. Review the information in the report. Consider the various metrics listed below. In addition, consider how the combinations of various key metrics can indicate the problem type. For example, the combination of the following four key metrics can indicate which type of problem is causing the SQL slowdown. You can then use additional metrics to further diagnose the problem.

Table 20. Key indicators for diagnosing an SQL slowdown problem

TOTAL_CPU _TIME	TOTAL_WAIT _TIME	TOTAL _SECTION _PROC_TIME	CLIENT_IDLE _WAIT_TIME	Indication
Normal	Normal	Normal	Higher than normal	No problem with the data server. The slowdown is localized upstream from the data server.
Higher than normal	Normal	Normal	Normal	The slowdown is occurring either before or after statement execution.
Higher than normal	Normal	Higher than normal	Normal	The slowdown is localized in statement processing.
Normal	Higher than normal	Normal	Normal	The slowdown is due to requests spending more time waiting.

3. Determine whether the slowdown is located in the data server, upstream in the client, in client-server communications, or in routine processing.

Table 21. Monitor elements used to determine the problem location

Metric	Interpretation of abnormal values
TOTAL_CPU_TIME	Higher CPU usage is a good indicator that the problem is located within data server processing. However, not all problems within data server processing result in higher processing usage. For example, wait-related problems, such as lock waits, might not result in higher processing resource usage.
Total request processing time (TOTAL_RQST_TIME-TOTAL_WAIT_TIME)	The data server is spending more time processing because it is either processing more requests or it is taking longer to process requests.
TOTAL_WAIT_TIME / TOTAL_RQST_TIME	Requests are spending more time waiting for a resource needed before processing can proceed.

Table 21. Monitor elements used to determine the problem location (continued)

Metric	Interpretation of abnormal values
CLIENT_IDLE_WAIT_TIME	An increase in this metric indicates a higher elapsed time between client requests, that is, between statements in a unit of work or between units of work. This means there is a higher proportion of time on an active connection where DB2 is waiting for the client to send its next request. A high value for this metric could explain an overall apparent slow down in end user response time. A common reason for high client idle wait time is that the user is away from the terminal or thinking. Alternately, client system might be experiencing higher load which impacts its ability to make requests to the data server.
TCPIP_SEND_WAIT_TIME TCPIP_RECV_WAIT_TIME	An increase in either of these metrics indicates increased time spent on waiting during client-server communications.
TOTAL_ROUTINE_INVOCATIONS, TOTAL_ROUTINE_PROC_TIME	Higher values for routine processing can indicate the slowdown is located in routine processing, such as a stored procedure. This processing could be related to SQL statements, to procedure SQL, or host programming languages used in a routine.

4. If the slowdown is caused by a resource bottleneck, determine whether it is located in the input-output processing or in system processing.

Table 22. Monitor elements used to determine if there is a computing resources bottleneck

Metric	Interpretation of abnormal values
TOTAL_CPU_TIME	The problem is located within data server processing. However, not all problems within data server processing result in higher processing resource usage. For example, wait-related problems, such as lock waits might not result in higher processor usage.
POOL_READ_TIME or POOL_WRITE_TIME	The data server is spending more time waiting for the storage system to complete data server requests. The increase may be due to either a greater number of reads or writes, or to individual reads or writes taking longer to execute.

5. Determine whether there are excessive wait requests. Check for common reasons that requests spend time waiting, including waiting for locks, input/output, or something else.

Table 23. Monitor elements used to determine whether requests are spending a long time waiting

Metric	Interpretation of abnormal values
TOTAL_WAIT_TIME / TOTAL_RQST_TIME	Requests are spending more time waiting for a resource needed before processing can proceed.
POOL_READ_TIME or POOL_WRITE_TIME	The data server is spending more time waiting for the storage system to complete data server requests. The increase may be due to either a greater number of reads or writes, or to individual reads or writes taking longer to execute.
LOCK_WAIT_TIME	Requests are spending more time waiting for locks.
WLM_QUEUE_TIME_TOTAL	Requests are waiting because they are queued by the DB2 workload manager.
AGENT_WAIT_TIME	Requests are waiting for an available agent.

6. Determine whether there is an access plan problem.

Table 24. Monitor elements used to determine whether there is an access plan problem

Metric	Interpretation of abnormal values
TOTAL_SECTION_PROC_TIME	Access plan executions are slowed down, assuming the number of executions has not changed.

Identifying the scope of an SQL slowdown

Determine whether the SQL slowdown affects all or most of the workload, or whether it is limited to SQL statements issued from a particular connection, workload, or service class. It is also possible that a slowdown is limited to a particular database member.

About this task

Use the information in the Summary report to confirm that an SQL slowdown is occurring. If it is, use this information to determine whether it affects all or most of the workload, or whether it is limited to particular connections, workloads, DB2 service classes, or database members.

Procedure

To determine the scope of SQL statements involved:

1. Generate the Summary report, as described in “Determining the problem category” on page 227 task.
2. Check the metrics listed in the Table 25 on page 231 table for one or more of the following: each connection, each DB2 service class, and each DB2 workload.

   Compare the calculated values to their normal historical values.In "Part 2 - Application performance drill down" section of the Summary report, review the following sections:

   - "Application performance by connection"
   - "Application performance by service class"
   - "Application performance by workload"

If metrics for one or a few connections (or workloads or service classes) are worse then normal, then the slowdown is limited to statements from that particular source. Further investigation should focus on that subset of SQL statements.

If metrics are worse for most or all connections (or DB2 service classes or DB2 workloads), then the problem is a general system slowdown. A general system slowdown most likely cannot be further diagnosed by collecting and analyzing individual SQL statements.

3. Refer to "Part 3 - Member level information" of the Summary report and use the information to calculate the following metrics for each database member:

Table 25. Monitor elements used to identify the scope of SQL slowdown

Metric	Interpretation of abnormal results
TOTAL_APP_COMMITS	The number of commit statements issued by the client application is the most common measure of throughput. This number is measured either since metrics have been collected or, if you specified an interval for the report, during that interval. A value that is lower than normal is a definitive indicator that users are experiencing a performance slowdown. The other metrics in this table provide confirmation as well as some indication about the nature of the problem.
TOTAL_CPU_TIME per request TOTAL_CPU_TIME /APP_RQSTS _COMPLETED_TOTAL TOTAL_WAIT_TIME percentage (TOTAL_WAIT_TIME / TOTAL _RQST_TIME) * 100	Slow downs that occur within the data server usually have one or both of the following symptoms: • Higher processing resource usage • Higher percentage of waiting time
ROWS_READ + ROWS_MODIFIED	The volume of rows processed through read and update operations is a convenient, approximate measure of how much work the data server is performing. A value that is higher than normal suggests that access plans have become more resource intensive and costly to execute.
TOTAL_CPU_TIME per request TOTAL_CPU_TIME/RQSTS _COMPLETED_TOTAL	Individual requests have become more expensive to process, which may be a cause of the slowdown being investigated.
RQSTS_COMPLETED_TOTAL	This member is working harder because it is processing more requests.
TOTAL_WAIT_TIME % (TOTAL_WAIT_TIME / TOTAL_ RQST_TIME) * 100	Requests on this member are spending more time waiting for a resource needed before processing can proceed.
Input and output wait time (POOL_READ_TIME + POOL_WRITE_TIME + DIRECT_READ_TIME + DIRECT_WRITE_TIME) / seconds in the monitoring interval	Requests on this member are spending more time waiting for input and output requests.

4. Compare the calculated metrics for each database member to the historical values of the calculated metrics for each database member.

If the metrics are worse then normal for particular database members, then those database members are most likely affected by the slowdown.

Reviewing top SQL statements

To determine which SQL statements are involved in the performance slowdown, review the lists of statements ranked by metrics such as **cpu_time** or **rows_read**. These lists are included in the Current SQL and Package Cache reports.

About this task

The top-ranked statements among those currently running can be determined from the Current SQL report. The top-ranked statements from recent history can be determined from the Package Cache report.

Procedure

1. Check the top-ranked statements currently running.
   a. Call the routine MONREPORT.CURRENTSQL to generate the Current SQL report. Review the "Part 2 - Overall ranking of activities" section and the "Part 1 - Summaries by 'top' metrics" section of the report. Part 1 shows several lists of top-ranked statements, in the following sections:
      - "Top 10 current activities by TOTAL_CPU_TIME"
      - "Top 10 current activities by ROWS_READ"
      - "Top 10 current activities by DIRECT READS + DIRECT WRITES"

      Each statement in a list is uniquely identified by ACTIVITY_ID, UOW_ID, and APPLICATION_HANDLE.
   b. Determine whether one or more of the statements might be the explanation for the slowdown. Review statements that fall into one or more of the following categories:
      - Statements with reported values that are worse than the historical normal values for the system.
      - Statements that appear in multiple top-ranked lists.
   c. Identify the statements involved. Use the "Part 3 - Complete statement text for activities" section to see the exact wording of each statement identified by the unique combination of ACTIVITY_ID, UOW_ID, and APPLICATION_HANDLE.
   d. For more information about a statement, including whether it was submitted by a routine, call the MON_GET_ACTIVITY_DETAILS table function, for the statements identified by the combination of ACTIVITY_ID, UOW_ID, and APPLICATION_HANDLE.
2. Check the top-ranked statements from recent history.
   a. Call the routine MONREPORT.PKGCACHE to generate the Package Cache report. Review the "Part 1 - Summaries by 'top' metrics" section. This section shows several lists of top-ranked statements, overall and for each execution:
      - Top 10 statements by I/O wait time
      - Top 10 statements by I/O wait time per exec
      - Top 10 statements by number of executions (If no member was specified, this value is based on the **num_coord_exec_with_metrics** monitor element. If a member was specified, this value is based on the **num_exec_with_metrics** monitor element.)
      - Top 10 statements by ROWS_READ + ROWS_MODIFIED
      - Top 10 statements by ROWS_READ + ROWS_MODIFIED per exec

- Top 10 statements by TOTAL_ACT_WAIT_TIME
- Top 10 statements by TOTAL_ACT_WAIT_TIME per exec
- Top 10 statements by TOTAL_CPU TIME per exec
- Top 10 statements by TOTAL_CPU_TIME

Each statement is identified with a number that is unique within the report and can be cross-referenced across the different sections and summary lists.

b. Review the report to determine whether one or more of the statements might be the explanation for the slowdown. Review statements that fall into one or more of the following categories:
   - Statements with reported values that are worse than the historical normal values for the system.
   - Statements that appear in multiple top-ranked lists.
   - Statements with a small problem that is being magnified because the statement is executed with a high frequency. Use the metrics summed across executions to identify if statements are causing problems cumulatively.

   Use this information in conjunction with any prior diagnosis information. For example, if you have already determined that there is a global increase in processing time, then review the TOTAL_CPU_TIME lists first. If the Summary report indicates that there has been an increase in wait times, then refer to the lists of statements by I/O wait time.

c. Identify the statement executions involved in the slowdown. Use the "Part 2 - EXECUTABLE_IDs for statements in Part 1" section of the report to find the executable ID for each statement.

d. For more information about a statement's execution history, call the MON_GET_PKG_CACHE_STMT table function.

```
DB2 SELECT STMT_TEXT FROM TABLE(MON_GET_PKG_CACHE_STMT)
  (null, x'EXECUTABLE_ID', ,))
```

Reviewing all work running on the system

When you review all the work currently running on the system, you can check for patterns that might identify the problem category or the SQL statements involved in the slowdown.

About this task

Use the Current Applications report to review all applications that are currently running on the system.

The report presents the current instantaneous state of processing of unit of work agents, and activities for each connection. The report starts with a summary section showing the number of current connections and activities and a series of summaries, such as current units of work by workload occurrence state.

Procedure

1. Call the routine MONREPORT.CURRENTAPPS to generate the Current Applications report.

   ```
   call monreport.currentapps;
   ```

2. Review the "Part 1 - Summary of application processing" section.
   a. Determine whether there is an abnormal proportion of current activities in a state associated with a slowdown, especially a wait state. For example,

determine whether a large proportion of Event Object:Event Type records display the LOCK:ACQUIRE value. This indicates that many activities may be waiting to acquire a lock.

b. Determine which users or applications are currently running. What has been the history of slowdowns in the past for the users or applications that are currently running? Have some of these users or applications been subject to particular types of slowdowns in the past?

3. Optional: Rerun the report multiple times, every few seconds, and look for whether requests seem to be progressing.
   - In "Part 2 - Details by connection" section, review the "Agents for this connection" section for the connections where you suspect a problem. Determine whether the REQUEST_TYPE values are changing. This indicates that old requests are not completing and new requests are not starting.
   - In "Part 2 - Details by connection" section, review the "Activities for this connection" section for the connections where you suspect a problem. Determine whether the ACTIVITY_STATE values are changing over time. If they are not, this indicates a problem.
   - In "Part 2 - Details by connection" section, review the "Agents for this connection" section for the connections where you suspect a problem. Determine whether the EVENT_OBJECT:EVENT_TYPE values are progressing. If they are not, this indicates a problem.

Collecting statement text and the details of the SQL statements involved

Troubleshooting an SQL performance slowdown incident often involves collecting statement text and other details of suspect statements for further investigation. This step occurs near the end of the initial troubleshooting phase when suspect statements have been identified from monitoring reports.

About this task

The MON_GET_PKG_CACHE_STMT table function is typically the best place to obtain SQL statement information. It contains the most information about the largest number of SQL statements.

Procedure

To obtain the statement text and other details for a specific SQL statement:

1. Review the Package Cache report, as described in "Reviewing top SQL statements" on page 232 task, to obtain the EXECUTABLE_ID for the statement.
2. Call the MON_GET_PKG_CACHE_STMT table function as follows:

   ```
   DB2 SELECT STMT_TEXT FROM TABLE(MON_GET_PKG_CACHE_STMT)
     (null, x'EXECUTABLE_ID', ,))
   ```

   where *EXECUTABLE_ID* is the EXECUTABLE_ID you obtained for the statement from the Package Cache report.
3. Review the output of the MON_GET_PKG_CACHE_STMT table function to find out additional information about the SQL statement. Some details provided in the output include the following:
   - The **stmt_text** monitor element provides the SQL statement text.
   - The **section_type** monitor element indicates whether the statement is static or dynamic.

4. Optionally, you can use other DB2 interfaces to collect additional SQL statement information.
   - You can use the MON_GET_ACTIVITY_DETAILS table function to find out details about the SQL statement. For example, the **routine_id** monitor element indicates whether the statement was issued from a routine.
   - You can use the MON_GET_CONNECTION_DETAILS table function to find details about the submitter of the SQL statement, including **appl_id**, **appl_name**, **system_auth_id**, and **client_userid** monitor elements.
   - The explain facility provides detailed information about the access plan that the optimizer choose for the SQL statement. Access the explain facility by using the **db2expln** or **db2exfmt** commands.
   - You can also use various event monitors, such as the package cache event monitor, to continue collecting information about the SQL statement execution.

Chapter 6. Determining the date a database object was last used

The last date that an object was used is indicated by the last referenced date (also referred to as the last used date). The last referenced date is available for indexes, packages, tables, table data partitions, and materialized query tables (MQTs). You can use the last referenced date to identify objects which have not been used for an extended period of time and which might be considered as candidates for removal.

The last referenced date is stored in the LASTUSED column of the corresponding catalog table for the object and accessible through the catalog view on the table. Usage information in the catalogs is updated by an engine dispatchable unit (EDU), called **db2lused** (the LASTUSED daemon), that runs on the database catalog partition. Every 15 minutes, the LASTUSED daemon gathers usage information for all objects across all partitions and updates the LASTUSED column in the corresponding catalog tables to write the information to disk. At most, the catalog entry for a given object is updated once per day, which means the same object will not be checked again until a 24 hour interval has passed. The 15 minute interval was chosen to minimally affect performance on the database server and is not user configurable. The updates to the last referenced date are performed asynchronously and, therefore, object access is not immediately recorded in the catalogs.

Note: If the corresponding row in a catalog table is locked, an update of usage information might be deferred until the next 15 minute collection interval. Also, when a database is deactivated, any usage information that was not gathered by the LASTUSED daemon prior to deactivation (for example, any objects accessed for the first time since the last poll was done by the daemon) cannot be written to disk. Explicitly activate the database for this feature to behave as expected.

The last referenced date is of interest when an object has not been used for an extended period of time (for example, several months). The last referenced date is useful in the following cases:

- Tables and table data partitions: can help to identify opportunities to reclaim unused space
- Indexes: can help to identify opportunities to reclaim unused space, avoid unnecessary inserts and maintenance, and can improve compile time by reducing the number of choices for an index to consider
- Packages: can help to detect unused package versions which can be freed
- MQTs: can help to detect unused MQTs, to reclaim unused space, or help to investigate and understand why an MQT is not being used

The following examples describe some specific scenarios in which the last referenced date can be useful:

- To identify opportunities to save space and maintenance overhead, you can examine last used information for indexes every year by checking the LASTUSED column in the SYSCAT.INDEXES catalog view. If an index has not been used in the last year, the index can be considered as a candidate for being dropped. The final decision to drop an index remains under your control because there might be circumstances in which dropping an index is not desired. For example, you might have a table which is known to be accessed only under emergency or infrequent cases where fast access is critical, or the index for a

© Copyright IBM Corp. 1993, 2012

table might be unique and used to enforce the uniqueness constraint even though it is never explicitly used. The last used date information can be used as an aid in making decisions to remove indexes.

- Your company has internal applications that were deployed on the database and were either replaced or are no longer in use after a period of months or years. The retired applications have been identified as opportunities to save space. The last used date information can be used to identify database objects that are no longer in use and were not cleaned up after an application was retired. For example, these database objects might be tables storing values used to populate a GUI. The last used date for these tables can be found in the LASTUSED column of the SYSCAT.TABLES catalog view and this date can be used as a starting point in the investigation of table objects that can be removed to reclaim space.

For additional information about the LASTUSED column of the catalog view for a specific database object, particularly which operations result in an update, see the following topics:

- SYSCAT.DATAPARTITIONS catalog view
- SYSCAT.INDEXES catalog view
- SYSCAT.PACKAGES catalog view
- SYSCAT.TABLES catalog view

Chapter 7. Time-spent monitor elements

Time-spent monitor elements track how time is spent in the system. You can query them to see where time is spent waiting, or performing different types of processing. You can also view the elapsed time spent in a particular system component.

Figure 6 shows an example of one way that you can view the relative time spent in waits and processing times for a request.

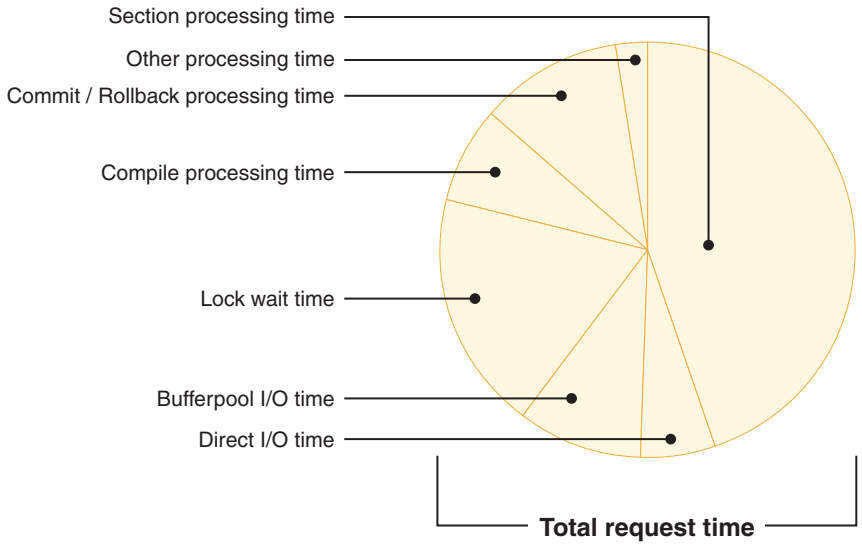


Figure 6. How time-spent metrics can provide an overall view of where time is spent in the system. Time is broken down into time spent waiting (lock wait time, buffer pool I/O time, direct I/O time, and time spent doing actual processing.

There are three ways that the database manager monitors time spent in the system:

- Wait times
- Component processing times
- Component elapsed times.

Wait times

Wait-time monitor elements reflect the time that the database manager spends waiting on something before it can proceed with processing. Some examples include time spent waiting for the following:

- Incoming client requests
- Locks on objects to be released
- Writing to the diagnostic log
- Reading from or writing to the buffer pool.

Examples of monitor elements that track wait time include **lock_wait_time** and **pool_read_time**.

Component processing times

These times represent the time spent doing actual processing within a specific logical component of the database. Some examples include time spent doing the following:

© Copyright IBM Corp. 1993, 2012

- Committing or rolling back transactions
- Performing database reorganizations
- Compiling SQL
- Loading data
- Performing runstats operations.

Examples of monitor elements that track component processing time include **total_compile_proc_time** and **total_commit_proc_time**.

Component elapsed times

Component elapsed times reflect the total amount of elapsed time spent within a logical component of the database. They include *both* processing time and various types of wait times that might be incurred during that overall stage of processing. For example, the overall time spent doing a commit includes actual commit processing and might also include various types of wait times, such as time incurred waiting for I/O operations or log file operations to complete.

Note: Elapsed time is not the same as elapsed time as measured on a clock; if the overall time spent was split among multiple threads, the time spent in each thread is represented in this number.

Some examples of how you can use component times include:

- Learning where relatively costly processing is taking place for a given workload (for example SQL compilation, as compared to query execution)
- Determining whether the cost of a specific component area can be attributed to actual processing, or whether wait time plays a significant role in reducing throughput
- Understanding the cost of a particular component area (for example, rollback processing) in the context of the overall time spent in the system.

Examples of monitor elements that track overall component times include **total_compile_time** and **total_commit_time**

You can query component processing times and wait times to get a breakdown of specific wait times relative to processing times. Figure 6 on page 239 is an example of how these two types of time-spent metrics can be viewed relative to one another.

While component elapsed times cannot be used to obtain a breakdown of specific types of wait times (for example, lock waits, I/O-related waits), they do provide an alternative view that you can use to view processing times relative to the overall time spent in a given logical database component. An example would be examining the ratio of actual processing time for table or index reorganizations (**total_reorg_proc_time**), to the overall elapsed time spent performing reorgs (**total_reorg_time**), which could include time spent on a variety of miscellaneous processing and waits not directly related to the reorg itself.

Hierarchy of time-spent monitor elements

Many time-spent monitor elements rollup to more general monitor elements. For example, individual wait time elements, such as that for the time spent waiting to receive the next buffer from a table queue (**fcm_tq_recv_wait_time**) and time spent waiting for an FCM reply message (**fcm_message_recv_wait_time**) are both

included in overall **fcm_recv_wait_time** element. The hierarchical organization of time-spent monitor element makes it possible to choose the element with the most appropriate level of specificity for your needs.

Dimensions and perspectives for viewing time-spent monitor elements

There are different ways that you can look at the hierarchies of time-spent monitor elements. One way is to look at them from the viewpoint of the system as a whole; you can also look at them in the context of specific activities within the system.

The system-level view or *dimension* includes elements that you can use to see what the system is doing as a whole. You can also use elements in the system dimension to view time-spent information as it relates to specific workloads.

The activity-level view or dimension includes elements that you can use to see where the system is spending time on specific activities, such as the execution of SQL statements. All monitor elements in the activity dimension are included in the higher-level system dimension.

Within each of these two dimensions are two different *perspectives* you can use to look at time-spent monitor elements:

1. Component processing times as compared to wait times
2. Component elapsed times as compared to component processing times.

In first perspective, the values for wait time elements are independent of and complementary to the values for component processing time elements. If you add the sum of all of the reported wait times to the sum of all component processing times, the resulting value would be very close to the value reflected by the **total_rqst_time** monitor element. Any minor difference between the two values is the result of a small amount of miscellaneous component processing time that is not currently tracked in any monitor element.

With the second of these two perspectives, component elapsed times are a superset of component processing times. In other words, for a logical component of the database such as that that performs commits, the total amount of commit processing time (**total_commit_proc_time**) is *included in* the overall elapsed time for the commit (**total_commit_time**). The difference between the total elapsed time and the total processing time is the made up of miscellaneous wait or processing times that are not individually tracked as part of the component elapsed time monitor element.

Viewing component elapsed times relative to wait times is *not* meaningful, because component elapsed times already include wait times incurred as part of the elapsed time in that part of the system. If you were to create a pie chart consisting of both component elapsed times and wait times, it would not be an accurate representation of the time spent in your system because you would actually be double-counting the various types of wait times.

The sections that follow describe the various dimensions (system, activity) and perspectives (component processing/wait times, component elapsed/component processing times) from which you can view time-spent monitor elements.

Note: Not all time-spent elements are reported through all interfaces. For example, the **client_idle_wait_time** monitor element is only applicable to system-level

interfaces like the MON_GET_SERVICE_SUBCLASS table function. Refer to the reference topic for each monitor element for a list of the interfaces that report the element.

- "Time-spent monitor element hierarchy - System dimension"
- "Time-spent monitor element hierarchy - Activity dimension" on page 245

Time-spent monitor element hierarchy - System dimension

Figure 7 on page 243 shows an overall view of how both the wait and component processing time monitor elements relate to one another, as viewed from the system dimension

- "client_idle_wait_time - Client idle wait time monitor element" on page 557
- "total_rqst_time - Total request time monitor element" on page 1023
  - "total_wait_time - Total wait time monitor element" on page 1038
    - "agent_wait_time - Agent wait time monitor element" on page 511
    - "wlm_queue_time_total - Workload manager total queue time monitor element" on page 1060
    - "lock_wait_time - Time waited on locks monitor element" on page 734
    - "log_buffer_wait_time - Log buffer wait time monitor element" on page 740
    - "log_disk_wait_time - Log disk wait time monitor element" on page 741
    - "tcpip_recv_wait_time - TCP/IP received wait time monitor element" on page 976
    - "tcpip_send_wait_time - TCP/IP send wait time monitor element" on page 979
    - "ipc_recv_wait_time - Interprocess communication received wait time monitor element" on page 705
    - "ipc_send_wait_time - Interprocess communication send wait time monitor element" on page 708
    - "fcm_recv_wait_time - FCM received wait time monitor element" on page 648[1]
      - "fcm_tq_recv_wait_time - FCM table queue received wait time monitor element" on page 656[1]
      - "fcm_message_recv_wait_time - FCM message received wait time monitor element" on page 641[1]
    - "fcm_send_wait_time - FCM send wait time monitor element" on page 652[1]
      - "fcm_tq_send_wait_time - FCM table queue send wait time monitor element" on page 659[1]
      - "fcm_message_send_wait_time - FCM message send wait time monitor element" on page 645[1]
    - "audit_subsystem_wait_time - Audit subsystem wait time monitor element" on page 534
    - "audit_file_write_wait_time - Audit file write wait time monitor element" on page 532
    - "diaglog_write_wait_time - Diagnostic log file write wait time monitor element" on page 618
    - "pool_read_time - Total buffer pool physical read time monitor element" on page 833
    - "pool_write_time - Total buffer pool physical write time monitor element" on page 847
    - "direct_read_time - Direct read time monitor element" on page 622
    - "direct_write_time - Direct write time monitor element" on page 627
    - "ida_send_wait_time - Time spent waiting to send data monitor element" on page 689
    - "ida_recv_wait_time - Time spent waiting to receive data monitor element" on page 685
  - "total_compile_proc_time - Total compile processing time monitor element" on page 998
  - "total_implicit_compile_proc_time - Total implicit compile processing time monitor element" on page 1004
  - "total_routine_user_code_proc_time - Total routine user code processing time monitor element" on page 1020
  - "total_section_proc_time - Total section processing time monitor element" on page 1027
    - "total_section_sort_proc_time - Total section sort processing time monitor element" on page 1028
    - *Other*[2]
  - "total_commit_proc_time - Total commits processing time monitor element" on page 995
  - "total_rollback_proc_time - Total rollback processing time monitor element" on page 1014
  - "total_runstats_proc_time - Total runtime statistics processing time monitor element" on page 1025
  - "total_reorg_proc_time - Total reorganization processing time monitor element" on page 1011
  - "total_load_proc_time - Total load processing time monitor element" on page 1006
  - *Other*[3]

[1]These FCM-related wait times do not yield meaningful information when aggregated across members. See "Wait times for FCM communications" on page 246 for more information.

[2]Includes miscellaneous processing and wait times that are not specifically related to this component. For example, **total_section_proc_time** is comprised of **total_section_proc_sort_time** time spent performing activities like table and index scans or joins.

[3] Includes a small number of miscellaneous types of time spent (processing and wait times) that are not currently monitored.

Figure 7. Wait and component processing time spent monitor elements - System dimension. The values for indented monitor elements are included in the element that precedes it in the next-highest level of the hierarchy.

The preceding hierarchy shows how the monitor elements for wait times and component processing times roll-up to the overall request time monitor element (**total_rqst_time**). In other words, the total request time comprises all wait times and all component processing times.

Figure 8 shows a detailed view of the monitor elements for time spent in various component areas. Each component time is represented by two different monitor elements:

- One that reports the total amount of *processing time* in a component or stage of processing
- One that reports on the *overall elapsed time* spent in the component. This overall time includes the processing time for the component as well as any miscellaneous other processing or wait times that may be involved.

- "total_rqst_time - Total request time monitor element" on page 1023
  - "total_compile_time - Total compile time monitor element" on page 999
    - "total_compile_proc_time - Total compile processing time monitor element" on page 998
    - *Other*[1]
  - "total_implicit_compile_time - Total implicit compile time monitor element" on page 1005
    - "total_implicit_compile_proc_time - Total implicit compile processing time monitor element" on page 1004
    - *Other*[1]
  - "total_routine_user_code_time - Total routine user code time monitor element" on page 1021
    - "total_routine_user_code_proc_time - Total routine user code processing time monitor element" on page 1020
  - "total_section_time - Total section time monitor element" on page 1033
    - "total_section_sort_time - Total section sort time monitor element" on page 1030
      - "total_section_sort_proc_time - Total section sort processing time monitor element" on page 1028
      - *Other*[1]
    - *Other*[1]
  - "total_commit_time - Total commit time monitor element" on page 996
    - "total_commit_proc_time - Total commits processing time monitor element" on page 995
    - *Other*[1]
  - "total_rollback_time - Total rollback time monitor element" on page 1015
    - "total_rollback_proc_time - Total rollback processing time monitor element" on page 1014
    - *Other*[1]
  - "total_runstats - Total runtime statistics monitor element" on page 1024
    - "total_runstats_proc_time - Total runtime statistics processing time monitor element" on page 1025
    - *Other*[1]
  - "total_reorg_time - Total reorganization time monitor element" on page 1012
    - "total_reorg_proc_time - Total reorganization processing time monitor element" on page 1011
    - *Other*[1]
  - "total_load_time - Total load time monitor element" on page 1007
    - "total_load_proc_time - Total load processing time monitor element" on page 1006
    - *Other*[1]
  - *Other*[2]

[1]Includes one or more different types of wait times.

[2]Includes a small number of miscellaneous types of time spent (processing and wait times) that are not currently monitored.

Figure 8. Component time-spent monitor elements - System dimension. The values for indented monitor elements are included in the element that precedes it in the next-highest level of the hierarchy.

Time-spent monitor element hierarchy - Activity dimension

In addition to being able to view time-spent monitor elements from a system-level perspective, you can also view time-spent monitor elements from the activity dimension. Figure 9 shows the monitor elements that you can view for activities from the perspective of wait times as compared to component processing times.

- "stmt_exec_time - Statement execution time monitor element" on page 931
  - "total_act_wait_time - Total activity wait time monitor element" on page 989 [1]
    - "lock_wait_time - Time waited on locks monitor element" on page 734
    - "log_buffer_wait_time - Log buffer wait time monitor element" on page 740
    - "log_disk_wait_time - Log disk wait time monitor element" on page 741
    - "fcm_recv_wait_time - FCM received wait time monitor element" on page 648[2]
      - "fcm_tq_recv_wait_time - FCM table queue received wait time monitor element" on page 656[2]
      - "fcm_message_recv_wait_time - FCM message received wait time monitor element" on page 641[2]
    - "fcm_send_wait_time - FCM send wait time monitor element" on page 652[2]
      - "fcm_tq_send_wait_time - FCM table queue send wait time monitor element" on page 659[2]
      - "fcm_message_send_wait_time - FCM message send wait time monitor element" on page 645[2]
    - "audit_subsystem_wait_time - Audit subsystem wait time monitor element" on page 534
    - "audit_file_write_wait_time - Audit file write wait time monitor element" on page 532
    - "diaglog_write_wait_time - Diagnostic log file write wait time monitor element" on page 618
    - "pool_read_time - Total buffer pool physical read time monitor element" on page 833
    - "pool_write_time - Total buffer pool physical write time monitor element" on page 847
    - "direct_read_time - Direct read time monitor element" on page 622
    - "direct_write_time - Direct write time monitor element" on page 627
    - "ida_send_wait_time - Time spent waiting to send data monitor element" on page 689
    - "ida_recv_wait_time - Time spent waiting to receive data monitor element" on page 685
  - "total_routine_non_sect_proc_time - Non-section processing time monitor element" on page 1017
    - "total_routine_user_code_proc_time - Total routine user code processing time monitor element" on page 1020
    - *Other*[3]
  - "total_section_proc_time - Total section processing time monitor element" on page 1027
    - "total_section_sort_proc_time - Total section sort processing time monitor element" on page 1028
    - *Other*[3]
  - *Other*[4]

[1]Does not include any wait time incurred by nested (child) activities executed by the statement.

[2]These FCM-related wait times do not yield meaningful information when aggregated across members. See "Wait times for FCM communications" on page 246 for more information.

[3]Includes miscellaneous processing times that are not specifically related to this component.

[4]Includes a small number of miscellaneous types of time spent (processing and wait times) that are not currently monitored. In addition, this time includes any processing and wait times incurred by child activities.

Figure 9. Wait and component processing time-spent monitor elements - Activity dimension. The values for indented monitor elements are included in the element that precedes it in the next-highest level of the hierarchy.

Figure 10 on page 246 shows the monitor elements that you can view for activities from the perspective of component elapsed times, which include component processing times.

- "stmt_exec_time - Statement execution time monitor element" on page 931
  - "total_section_time - Total section time monitor element" on page 1033
    - "total_section_sort_time - Total section sort time monitor element" on page 1030
      - "total_section_sort_proc_time - Total section sort processing time monitor element" on page 1028
      - *Other*[1]
    - *Other*[2]
  - "total_routine_time - Total routine time monitor element" on page 1018
    - "total_routine_non_sect_time - Non-section routine execution time monitor elements" on page 1017
      - "total_routine_user_code_time - Total routine user code time monitor element" on page 1021
      - *Other*[2]
    - *Other*[2]

[1]Includes one or more different types of wait times.

[2]Includes miscellaneous processing and wait times that are not specifically related to this component.

Figure 10. Component elapsed and component processing times - Activity dimension. The values for indented monitor elements are included in the element that precedes it in the next-highest level of the hierarchy.

Wait times for FCM communications

In a multi-partition database, or in an environment where there is intra-partition parallelism, the Fast Communications Manager (FCM) manages communication between different agents working on the same statement, whether those agents are on the same or on different members. All FCM communications involve the potential for wait times as one agent waits for work to be done by another, or for data to transferred from one to another.

FCM-related wait times do not necessarily indicate that processing is blocked across members; for a given statement, work might be proceeding in parallel or serially on sub-agents across members. The FCM-related wait times show the time that an agent is blocked on a single member waiting for another; however, work might very well be proceeding on the other member.

For example, Agent A on member 0 might be blocked waiting for Agent B on member 1 to read the data being sent to it. If Agent B is busy and does not receive the data from the table queue immediately, Agent A will only be allowed to send a limited amount of data before being forced to wait for an acknowledgement from Agent B before sending the remaining data. This wait time is counted as **fcm_tq_send_wait_time** by Agent A.

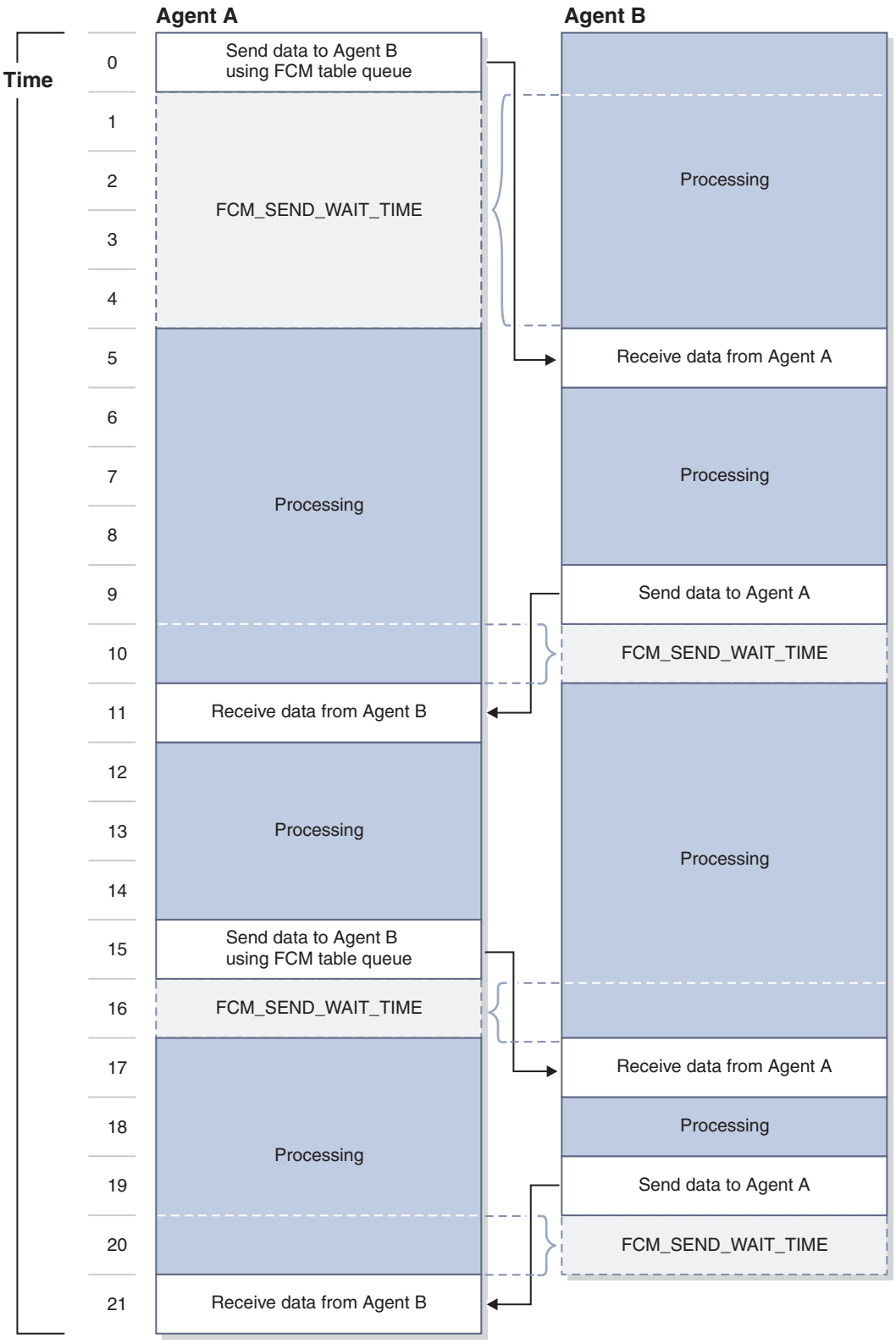


Figure 11. Wait times in FCM communications

Another scenario might involve an agent on one member dispatching a request to an agent on another member. **fcm_message_recv_wait_time** will be incurred if one of the following situations takes place:

- Agent A sends a lengthy request to agent B, and agent B is forced to wait for the full request to be received. In this case, agent B incurs **fcm_message_recv_wait_time**.
- Agent A sends a request to agent B, and awaits a reply from agent B. In this case, agent A incurs **fcm_message_recv_wait_time**.

fcm_message_send_wait_time will be incurred if one of the following situations takes place:

- Agent A sends a lengthy request to agent B, and is blocked for some reason. For example, Agent A might need to wait while the first portion of the request being sent is processed by the local FCM daemon. In this case, agent A incurs **fcm_message_send_wait_time**.
- Agent B sends a reply to a request from agent A. If agent B is for some reason blocked before the entire message can be sent, then agent B incurs **fcm_message_send_wait_time**.

Depending on what it is that you want to measure, it might be appropriate to subtract FCM wait times from the total time spent if you are aggregating metrics for time spent across multiple partitions.

Retrieving and working with time-spent monitor element data

Time-spent monitor element data can be used in almost unlimited ways. For example, you can automate the production of charts that show, at a glance, how time is spent in the system. Or you could use the data to track certain types of wait times in the system over a period of time.

The topics that follow provide some basic examples of how to use the time-spent monitor elements, as well as the table functions that you use to access the data they contain.

Seeing where time is spent across the system

You can use the time-spent monitor elements to get a view of where time is being spent in the system. Time spent monitor elements can be used to report on specific units of work, service subclasses, workloads or connections.

About this task

Once you retrieve the various monitor elements that report where time is being spent in the system, you can view them in several ways. At the most basic level, you can view the reported values as a list. You might want to use the values to create ratios, for example the ratio of lock wait time to total request time. Or you could use the values retrieved to create charts to help you visualize time spent monitor elements relative to another.

Notes:

- The values shown in the output for queries are for illustrative purposes only, and should not be construed as representative of what you might see in your own system.
- This task shows you how to retrieve specific time-spent monitor elements. You can also use new formatting functions introduced in Version 9.7 Fix Pack 1 to retrieve time spent monitor elements that meet specific criteria, such as those

with non-zero values, those within a certain range of values that you specify, or the top *n* monitor element (for example, the top five wait times). Example 4 illustrates how these functions work.

Procedure

1. First, determine what time-spent elements you are interested in. For example, you might want to look at total wait time as compared to the total request time for all connections in your system.
2. Formulate an SQL query that uses one of the monitoring table functions that retrieve the elements you are interested in. In this case, you can retrieve the **total_request_time** and the **total_wait_time** monitor elements for a connection using the MON_GET_CONNECTION table function:

   ```
   SELECT APPLICATION_HANDLE,
          TOTAL_WAIT_TIME,
          TOTAL_RQST_TIME
   FROM TABLE(MON_GET_CONNECTION(NULL,NULL))
   ```

   The output of the preceding query resembles the following (times are reported in milliseconds):

```
APPLICATION_HANDLE   TOTAL_WAIT_TIME      TOTAL_RQST_TIME
-------------------- -------------------- --------------------
                  39                  179                  269
                  78                    0                    0
                  51                  207                  316
                  77                    0                   21
                  50                 1014                 1408
                  40                  109                  351
                  79                   89                  167

  7 record(s) selected.
```

3. In this case, there are 7 application connections; you could use the results from the second and third columns to determine the percentage of time spent waiting for each application. For example, for application 50, compared to the total request time, the wait time spent by that application is $(1014 \div 1408) \times 100 \approx 72\%$.

Example

Example 1: Determining the average across all connections of the time spent waiting relative to overall request time.

This example is similar to the preceding one, except that the calculation of the average percentage wait time is done within the SQL:

```
WITH PCTWAIT AS (
     SELECT SUM(TOTAL_WAIT_TIME)AS WAIT_TIME,
            SUM(TOTAL_RQST_TIME)AS RQST_TIME
     FROM TABLE(MON_GET_CONNECTION(NULL,NULL)) AS METRICS)
     SELECT WAIT_TIME,
            RQST_TIME,
     CASE WHEN RQST_TIME > 0
     THEN DEC((FLOAT(WAIT_TIME))/FLOAT(RQST_TIME) * 100,5,2)
     ELSE NULL END AS WAIT_PCT FROM PCTWAIT
```

The results of running the preceding query would look something like this:

```
WAIT_TIME           RQST_TIME           WAIT_PCT
------------------- ------------------- --------
               1515                2439    62.11

  1 record(s) selected.
```

Example 2: Comparing total wait time against selected component processing times for a specific service subclass

This example shows how you can compare time spent in specific types of component processing with the time spent waiting:

```
SELECT SUM(TOTAL_WAIT_TIME) AS WAIT,
       SUM(TOTAL_COMPILE_PROC_TIME) AS COMPILE,
       SUM(TOTAL_IMPLICIT_COMPILE_PROC_TIME) AS IMP_COMPILE,
       SUM(TOTAL_SECTION_PROC_TIME) AS SECTION,
       SUM(TOTAL_COMMIT_PROC_TIME) AS COMMIT,
       SUM(TOTAL_REORG_PROC_TIME) AS REORG,
       SUM(TOTAL_RUNSTATS_PROC_TIME) AS RUNSTATS,
       SUM(TOTAL_ROLLBACK_PROC_TIME) AS ROLLBACK,
       SUM(TOTAL_LOAD_PROC_TIME) AS LOAD
 FROM TABLE(MON_GET_SERVICE_SUBCLASS( 'SYSDEFAULTUSERCLASS','SYSDEFAULTSUBCLASS',NULL))
```

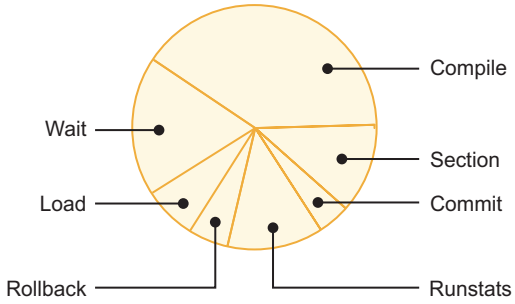
The results of the preceding query would look something like this (the output rows from the query have been split for presentation purposes):

```
WAIT                COMPILE             IMP_COMPILE         SECTION             COMMIT
------------------- ------------------- ------------------- ------------------- -------------------
                611                1931                   0                 395                  15

REORG               RUNSTATS            ROLLBACK            LOAD
------------------- ------------------- ------------------- -------------------
                  0                 432                  18                   0

  1 record(s) selected.
```

The numbers reported could be used to construct a pie chart to show the relative time spent waiting as compared to time spent in different stages of processing (component times of 0 are not included):



Example 3: View the ratio of total time spent as compared to processing time in different components

This example shows you how you can get an overview of the time spent doing work in different stages of processing (components), relative to the total time spent in that component. The following query computes the ratio (expressed as a percentage) of the time spent in actual processing as compared to the total elapsed time spent in a specific component.

```
WITH PCTPROC AS (
     SELECT SUM(TOTAL_SECTION_TIME) AS SECT_TIME, SUM(TOTAL_SECTION_PROC_TIME) AS SECT_PROC_TIME,
            SUM(TOTAL_COMPILE_TIME) AS COMP_TIME, SUM(TOTAL_COMPILE_PROC_TIME) AS COMP_PROC_TIME,
            SUM(TOTAL_IMPLICIT_COMPILE_TIME) AS IMP_C_TIME, SUM(TOTAL_IMPLICIT_COMPILE_PROC_TIME) AS IMP_C_PROC_TIME,
            SUM(TOTAL_COMMIT_TIME) AS COMMIT_TIME, SUM(TOTAL_COMMIT_PROC_TIME) AS COMMIT_PROC_TIME,
            SUM(TOTAL_ROLLBACK_TIME) AS ROLLBACK_TIME, SUM(TOTAL_ROLLBACK_PROC_TIME) AS ROLLBACK_PROC_TIME,
            SUM(TOTAL_RUNSTATS_TIME) AS RUNSTATS_TIME, SUM(TOTAL_RUNSTATS_PROC_TIME)AS RUNSTATS_PROC_TIME,
            SUM(TOTAL_REORG_TIME) AS REORG_TIME, SUM(TOTAL_REORG_PROC_TIME) AS REORG_PROC_TIME,
            SUM(TOTAL_LOAD_TIME) AS LOAD_TIME, SUM(TOTAL_LOAD_PROC_TIME) AS LOAD_PROC_TIME
     FROM TABLE(MON_GET_CONNECTION(NULL, -2)) AS METRICS)
     SELECT CASE WHEN SECT_TIME > 0
                 THEN DEC((FLOAT(SECT_PROC_TIME) / FLOAT(SECT_TIME)) * 100,5,1)
                 ELSE NULL END AS SECT_PROC_PCT,
            CASE WHEN COMP_TIME > 0
                 THEN DEC((FLOAT(COMP_PROC_TIME) / FLOAT(COMP_TIME)) * 100,5,1)
                 ELSE NULL END AS COMPILE_PROC_PCT,
            CASE WHEN IMP_C_TIME > 0
                 THEN DEC((FLOAT(IMP_C_PROC_TIME) / FLOAT(IMP_C_TIME)) * 100,5,1)
                 ELSE NULL END AS IMPL_COMPILE_PROC_PCT,
              CASE WHEN ROLLBACK_TIME > 0
                 THEN DEC((FLOAT(ROLLBACK_PROC_TIME) / FLOAT(ROLLBACK_TIME)) * 100,5,1)
                 ELSE NULL END AS ROLLBACK_PROC_PCT,
              CASE WHEN COMMIT_TIME > 0
                 THEN DEC((FLOAT(COMMIT_PROC_TIME) / FLOAT(COMMIT_TIME)) * 100,5,1)
                 ELSE NULL END AS COMMIT_PROC_PCT,
              CASE WHEN RUNSTATS_TIME > 0
                 THEN DEC((FLOAT(RUNSTATS_PROC_TIME) / FLOAT(RUNSTATS_TIME)) * 100,5,1)
                 ELSE NULL END AS RUNSTATS_PROC_PCT,
            CASE WHEN REORG_TIME > 0
                 THEN DEC((FLOAT(REORG_PROC_TIME) / FLOAT(REORG_TIME)) * 100,5,1)
                 ELSE NULL END AS REORG_PROC_PCT,
            CASE WHEN LOAD_TIME > 0
                 THEN DEC((FLOAT(LOAD_PROC_TIME) / FLOAT(LOAD_TIME)) * 100,5,1)
                 ELSE NULL END AS LOAD_PROC_PCT
       FROM PCTPROC
```

The query produces the following output:

```
SECT_PROC_PCT COMPILE_PROC_PCT IMPL_COMPILE_PROC_PCT ROLLBACK_PROC_PCT COMMIT_PROC_PCT RUNSTATS_PROC_PCT REORG_PROC_PCT LOAD_PROC_PCT
------------- ---------------- ---------------------- ----------------- ---------------- ------------------ --------------- --------------
         57.6              0.1                      -              96.9             95.6                0.0            71.1           84.6

  1 record(s) selected.
```

A graphical representation of this data might look something like what is shown in Figure 12:

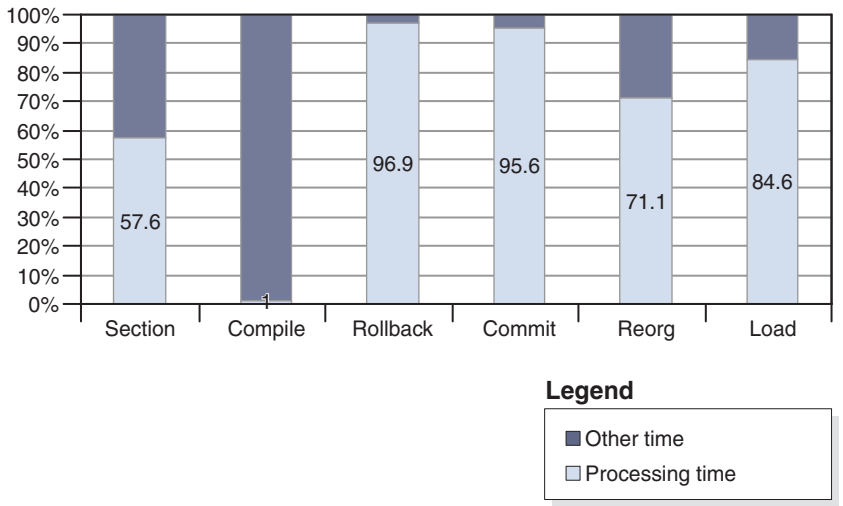


Figure 12. Component processing times as a percentage of overall time spent

Example 4: Ranking time-spent monitor elements

In the preceding examples, all monitor elements that are displayed are specified explicitly in the SQL for the query; each appears in its own column in the query results. However, there might be times when you do not know which time spent monitor elements you want to examine, such

as if you want to see the top ten wait-time monitor elements, or only the non-zero time-spent monitor elements.

Several table functions were added in DB2 Version 9.7 Fix Pack 1 that you can use to display monitor elements in a row-oriented format, where each element appears in a row by itself. The table functions you can use to do this have names of the form MON_FORMAT_XML_* _BY_ROW. These functions extract metrics from XML documents returned by certain monitoring interfaces. (See Chapter 3, "Interfaces that return monitor data in XML documents," on page 9 for more information.)
The MON_FORMAT_XML_* _BY_ROW functions are useful when you do not know which elements you want to view. For example, you might want to see the top 10 wait-time monitor elements for the workload named CLPWORKLOAD. To collect this information, you can create a statistics event monitor called DBSTATS (event_wlstats logical data group). Assuming you set up this event monitor to write to a table, it records metrics in a column called DETAILS_XML. Once the output table from the event monitor is populated with monitor data, you can construct a query that uses the MON_FORMAT_XML_WAIT_TIMES_BY_ROW function to extract the monitor elements you want to see:

```
SELECT SUBSTR(STATS.WORKLOAD_NAME,1,15) AS WORKLOAD_NAME,
       SUBSTR(METRICS.METRIC_NAME,1,30) AS METRIC_NAME,
       SUM(METRICS.TOTAL_TIME_VALUE) AS TOTAL_TIME_VALUE
FROM   WLSTATS_DBSTATS AS STATS,
       TABLE(MON_FORMAT_XML_WAIT_TIMES_BY_ROW(STATS.DETAILS_XML)) AS METRICS
WHERE  WORKLOAD_NAME='CLPWORKLOAD' AND (PARENT_METRIC_NAME='TOTAL_WAIT_TIME')
GROUP  BY WORKLOAD_NAME,METRIC_NAME
ORDER  BY TOTAL_TIME_VALUE DESC
FETCH FIRST 10 ROWS ONLY
```

Remember: Time spent monitor elements are organized into hierarchies. In this example, to avoid double-counting wait times, only the monitor elements that roll-up to **total_wait_time** are included (see the WHERE clause in the preceding SQL statement). Otherwise, total_wait_time itself would be included in the results, which includes several individual wait times.
The output that follows shows what the results of the preceding query might look like:

```
WORKLOAD_NAME   METRIC_NAME                    TOTAL_TIME_VALUE
--------------- ------------------------------ --------------------
CLPWORKLOAD     LOCK_WAIT_TIME                        15138541
CLPWORKLOAD     DIRECT_READ_TIME                       6116231
CLPWORKLOAD     POOL_READ_TIME                         6079458
CLPWORKLOAD     DIRECT_WRITE_TIME                       452627
CLPWORKLOAD     POOL_WRITE_TIME                         386208
CLPWORKLOAD     IPC_SEND_WAIT_TIME                      283172
CLPWORKLOAD     LOG_DISK_WAIT_TIME                      103888
CLPWORKLOAD     DIAGLOG_WRITE_WAIT_TIME                  78198
CLPWORKLOAD     IPC_RECV_WAIT_TIME                       15612
CLPWORKLOAD     TCPIP_SEND_WAIT_TIME                      3291

  10 record(s) selected.
```

Determining where time is spent during SQL statement execution

One example of retrieving time-spent information at the activity level is viewing time spent monitor elements for specific SQL statements. You can use the MON_GET_PKG_CACHE_STMT table function to retrieve this information.

About this task

This task shows an example of how to retrieve selected time-spent details for SQL statements in the package cache.

Note:

- The time-spent metrics reported for a given statement in the package cache are aggregates of the time-spent metrics for all executions of that statement.
- The values shown in the output for queries are for illustrative purposes only, and should not be construed as representative of what you might see in your own system.

Procedure

1. Formulate an SQL statement that uses the MON_GET_PKG_CACHE_STMT table function to retrieve information about statements in the package cache. For example, assume that you want to determine the total wait time relative to the total statement execution time. A query to retrieve this might look like this:

```
SELECT SUM(STMT_EXEC_TIME) AS TOTAL_EXEC_TIME,
       SUM(TOTAL_ACT_WAIT_TIME) AS TOTAL_WAIT_TIME,
       EXECUTABLE_ID
FROM TABLE(MON_GET_PKG_CACHE_STMT ( NULL, NULL, NULL, -2)) AS T
WHERE STMT_EXEC_TIME <> 0
GROUP BY EXECUTABLE_ID
ORDER BY TOTAL_EXEC_TIME DESC
```

2. Run the query. The results might look like the following:

```
TOTAL_EXEC_TIME      TOTAL_WAIT_TIME      EXECUTABLE_ID
-------------------- -------------------- --------------------------------------------------------------------------------
                9021                 9021 x'0100000000000000320000000000000000000000002002009111120320140000'
                3017                  372 x'0100000000000000300000000000000000000000002002009111115438062000'
                 591                    0 x'0100000000000000010000000000000000000000002002009111115252265000'
                 203                  192 x'0100000000000000270000000000000000000000002002009111115936750000'
                 142                    0 x'010000000000000002B000000000000000000000002002009111115944000000'
                 111                   48 x'0100000000000000070000000000000000000000002002009111115441359002'
                 108                   35 x'010000000000000000B000000000000000000000002002009111115441750000'
                  55                    0 x'010000000000000000D000000000000000000000002002009111115442062000'
                  50                    0 x'010000000000000000C000000000000000000000002002009111115441921000'
                  38                    0 x'0100000000000000260000000000000000000000002002009111115936609003'
                  35                    2 x'010000000000000000A000000000000000000000002002009111115441609000'
                  35                   35 x'0100000000000000130000000000000000000000002002009111115442593001'
                  33                    0 x'0100000000000000120000000000000000000000002002009111115442531000'
                  32                    0 x'0100000000000000240000000000000000000000002002009111115936578000'
                  29                    0 x'010000000000000000E000000000000000000000002002009111115442203000'
                  24                   23 x'0100000000000000040000000000000000000000002002009111115440640000'
                  24                    0 x'0100000000000000110000000000000000000000002002009111115442484003'
                  20                    0 x'0100000000000000030000000000000000000000002002009111120241828000'
                  15                    0 x'0100000000000000050000000000000000000000002002009111115440984000'
                  14                    0 x'0100000000000000080000000000000000000000002002009111115441437000'
                  13                   13 x'010000000000000000F000000000000000000000002002009111115442406001'
                   4                    0 x'0100000000000000100000000000000000000000002002009111115442484001'
                   3                    0 x'0100000000000000180000000000000000000000002002009111115442828000'
                   3                    3 x'010000000000000001F000000000000000000000002002009111115936515000'
                   3                    0 x'0100000000000000290000000000000000000000002002009111115943968001'
                   2                    0 x'0100000000000000150000000000000000000000002002009111115442656001'
                   2                    0 x'0100000000000000170000000000000000000000002002009111115442750000'
                   1                    0 x'0100000000000000160000000000000000000000002002009111115442734000'
                   1                    0 x'0100000000000000280000000000000000000000002002009111115937000001'
                   1                    0 x'01000000000000002A000000000000000000000002002009111115943984000'

  30 record(s) selected.
```

Results

At this point, you could use the MON_GET_PKG_CACHE_STMT table function again to retrieve the statement text for any statement you are particularly interested in. For example, the statement with the highest wait time above could be determined using the following query:

```
SELECT VARCHAR(STMT_TEXT, 80) AS STMT_TEXT
     FROM TABLE(MON_GET_PKG_CACHE_STMT(NULL,x'0100000000000000320000000000000000000000002002009111120320140000',NULL,-2))
     AS T
```

The output of the preceding query would look something like this:

```
STMT_TEXT
------------------------------------------------------------------------------
UPDATE EMPLOYEE SET BONUS=10000 WHERE PERF_RATING=1
  1 record(s) selected.
```

Chapter 8. Snapshot monitor

You can use the snapshot monitor to capture information about the database and any connected applications at a specific time. Snapshots are useful for determining the status of a database system. Taken at regular intervals, they are also useful for observing trends and foreseeing potential problems. Some of the data from the snapshot monitor is obtained from the system monitor. The data available from the system monitor is determined by system monitor switches.

The system monitor accumulates information for a database only while it is active. If all applications disconnect from a database and the database deactivates, then the system monitor data for that database is no longer available. You can keep the database active until your final snapshot has been taken, either by starting the database with the ACTIVATE DATABASE command, or by maintaining a permanent connection to the database.

Snapshot monitoring requires an instance attachment. If there is not an attachment to an instance, then a default instance attachment is created. An instance attachment is usually done implicitly to the instance specified by the DB2INSTANCE environment variable when the first database system monitor API is invoked by the application. It can also be done explicitly, using the ATTACH TO command. Once an application is attached, all system monitor requests that it invokes are directed to that instance. This allows a client to monitor a remote server by simply attaching to the instance on it.

In partitioned database environments, snapshots can be taken at any partition of the instance, or globally using a single instance connection. A global snapshot aggregates the data collected at each partition and returns a single set of values.

You can capture a snapshot from the CLP, from SQL table functions, or by using the snapshot monitor APIs in a C or C++ application. A number of different snapshot request types are available, each returning a specific type of monitoring data. For example, you can capture a snapshot that returns only buffer pool information, or a snapshot that returns database manager information. Before capturing a snapshot, consider if you need information from monitor elements that are under monitor switch control. If a particular monitor switch is off, the monitor elements under its control will not be collected.

Access to system monitor data: SYSMON authority

Users that are part of the SYSMON database manager level group have the authority to gain access to database system monitor data. System monitor data is accessed using the snapshot monitor APIs, CLP commands, or SQL table functions.

The SYSMON authority group replaces the DB2_SNAPSHOT_NOAUTH registry variable as the means to enable users without system administration or system control authorities to access database system monitor data.

Aside from SYSMON authority, the only way to access system monitor data using the snapshot monitor is with system administration or system control authority.

Any user that is part of the SYSMON group or has system administration or system control authority can perform the following snapshot monitor functions:

© Copyright IBM Corp. 1993, 2012

- CLP Commands:
  - GET DATABASE MANAGER MONITOR SWITCHES
  - GET MONITOR SWITCHES
  - GET SNAPSHOT
  - LIST ACTIVE DATABASES
  - LIST APPLICATIONS
  - LIST DCS APPLICATIONS
  - LIST UTILITIES
  - RESET MONITOR
  - UPDATE MONITOR SWITCHES
- APIs:
  - db2GetSnapshot - Get Snapshot
  - db2GetSnapshotSize - Estimate Size Required for db2GetSnapshot() Output Buffer
  - db2MonitorSwitches - Get/Update Monitor Switches
  - db2ResetMonitor - Reset Monitor
- Snapshot SQL table functions without previously running SYSPROC.SNAP_WRITE_FILE

Capturing database system snapshots using snapshot administrative views and table functions

Authorized users can capture snapshots of monitor information for a DB2 instance by using snapshot administrative views or snapshot table functions. The snapshot administrative views provide a simple means of accessing data for all database partitions of the connected database. The snapshot table functions allow you to request data for a specific database partition, globally aggregated data, or data from all database partitions. Some snapshot table functions allow you to request data from all active databases.

Before you begin

You must have SYSADM, SYSCTRL, SYSMAINT, or SYSMON authority to capture a database snapshot. To obtain a snapshot of a remote instance, you must first connect to a local database belonging to that instance.

About this task

While new snapshot table functions may be required in future releases if new monitor data is available, the set of snapshot administrative views will remain the same with new columns added to the view, making the administrative views a good choice for application maintenance over time.

Each snapshot view returns a table with one row per monitored object per database partition with each column representing a monitor element. Each table function returns a table with one row per monitored object for the specified partition. The column names of the returned table correlate with the monitor element names.

For example, a snapshot of general application information for the SAMPLE database is captured as follows using the SNAPAPPL administrative view:

```
SELECT * FROM SYSIBMADM.SNAPAPPL
```

You can also select individual monitor elements from the returned table. For example, the following statement returns only the **agent_id** and **appl_id** monitor elements:

```
SELECT agent_id, appl_id FROM SYSIBMADM.SNAPAPPL
```

Snapshot administrative views and table functions cannot be used in conjunction with either of the following:

- Monitor switches commands/APIs
- Monitor reset commands/APIs

This restriction includes:

- GET MONITOR SWITCHES
- UPDATE MONITOR SWITCHES
- RESET MONITOR

This limitation is due to the fact that such commands use an INSTANCE ATTACH, while snapshot table functions make use of DATABASE CONNECTs.

To capture a snapshot using a snapshot administrative view:

Procedure

1. To capture a snapshot using a snapshot administrative view:
   a. Connect to a database. This can be any database in the instance you need to monitor. To be able to issue an SQL query with a snapshot administrative view, you must be connected to a database.
   b. Determine the type of snapshot you need to capture. If you want to capture a snapshot for a database other than the currently connected database, or if you want to retrieve data from a single database partition, or global aggregate data, you need to use a snapshot table function instead.
   c. Issue a query with the appropriate snapshot administrative view. For example, here is a query that captures a snapshot of lock information for the currently connected database:

      ```
      SELECT * FROM SYSIBMADM.SNAPLOCK
      ```

2. To capture a snapshot using a snapshot table function:
   a. Connect to a database. This can be any database in the instance you need to monitor. To be able to issue an SQL query with a snapshot table function, you must be connected to a database.
   b. Determine the type of snapshot you need to capture.
   c. Issue a query with the appropriate snapshot table function. For example, here is a query that captures a snapshot of lock information about the SAMPLE database for the current connected database partition:

      ```
      SELECT * FROM TABLE(SNAP_GET_LOCK('SAMPLE',-1)) AS SNAPLOCK
      ```

      The SQL table functions have two input parameters:

      database name
      VARCHAR(255). If you enter NULL, the name of the currently connected database is used.

      partition number
      SMALLINT. For the database partition number parameter, enter the integer (a value between 0 and 999) corresponding to the database partition number you need to monitor. To capture a snapshot for the

currently connected database partition, enter a value of -1. To capture a global aggregate snapshot, enter a value of -2. To capture a snapshot from all database partitions, do not specify a value for this parameter.

Note:

1) For the following list of snapshot table functions, if you enter a NULL for the currently connected database, you will get snapshot information for all databases in the instance:
   - SNAP_GET_DB_V95
   - SNAP_GET_DB_MEMORY_POOL
   - SNAP_GET_DETAILLOG_V91
   - SNAP_GET_HADR
   - SNAP_GET_STORAGE_PATHS
   - SNAP_GET_APPL_V95
   - SNAP_GET_APPL_INFO_V95
   - SNAP_GET_AGENT
   - SNAP_GET_AGENT_MEMORY_POOL
   - SNAP_GET_STMT
   - SNAP_GET_SUBSECTION
   - SNAP_GET_BP_V95
   - SNAP_GET_BP_PART
2) The database name parameter does not apply to the database manager level snapshot table functions; they have only a parameter for database partition number. The database partition number parameter is optional.

Capturing database system snapshot information to a file using the SNAP_WRITE_FILE stored procedure

With the SNAP_WRITE_FILE stored procedure you can capture snapshots of monitor data and save this information to files on the database server and allow access to the data by users who do not have SYSADM, SYSCTRL, SYSMAINT, or SYSMON authority. Any user can then issue a query with a snapshot table function to access the snapshot information in these files. In providing open access to snapshot monitor data, sensitive information (such as the list of connected users and the SQL statements they have submitted to the database) is available to all users who have the execution privilege for the snapshot table functions. The privilege to execute the snapshot table functions is granted to PUBLIC by default. (Note, however, that no actual data from tables or user passwords can be exposed using the snapshot monitor table functions.)

Before you begin

You must have SYSADM, SYSCTRL, SYSMAINT, or SYSMON authority to capture a database snapshot with the SNAP_WRITE_FILE stored procedure.

About this task

When issuing a call to the SNAP_WRITE_FILE stored procedure, in addition to identifying the database and partition to be monitored, you need to specify a *snapshot request type*. Each snapshot request type determines the scope of monitor data that is collected. Choose the snapshot request types based on the snapshot

table functions users will need to run. The following table lists the snapshot table functions and their corresponding request types.

Table 26. Snapshot request types

Snapshot table function	Snapshot request type
SNAP_GET_AGENT	APPL_ALL
SNAP_GET_AGENT_MEMORY_POOL	APPL_ALL
SNAP_GET_APPL_V95	APPL_ALL
SNAP_GET_APPL_INFO_V95	APPL_ALL
SNAP_GET_STMT	APPL_ALL
SNAP_GET_SUBSECTION	APPL_ALL
SNAP_GET_BP_PART	BUFFERPOOLS_ALL
SNAP_GET_BP_V95	BUFFERPOOLS_ALL
SNAP_GET_DB_V95	DBASE_ALL
SNAP_GET_DETAILLOG_V91	DBASE_ALL
SNAP_GET_DB_MEMORY_POOL	DBASE_ALL
SNAP_GET_HADR	DBASE_ALL
SNAP_GET_STORAGE_PATHS	DBASE_ALL
SNAP_GET_DBM_V95	DB2
SNAP_GET_DBM_MEMORY_POOL	DB2
SNAP_GET_FCM	DB2
SNAP_GET_FCM_PART	DB2
SNAP_GET_SWITCHES	DB2
SNAP_GET_DYN_SQL_V95	DYNAMIC_SQL
SNAP_GET_LOCK	DBASE_LOCKS
SNAP_GET_LOCKWAIT	APPL_ALL
SNAP_GET_TAB_V91	DBASE_TABLES
SNAP_GET_TAB_REORG	DBASE_TABLES
SNAP_GET_TBSP_V91	DBASE_TABLESPACES
SNAP_GET_TBSP_PART_V91	DBASE_TABLESPACES
SNAP_GET_CONTAINER_V91	DBASE_TABLESPACES
SNAP_GET_TBSP_QUIESCER	DBASE_TABLESPACES
SNAP_GET_TBSP_RANGE	DBASE_TABLESPACES
SNAP_GET_UTIL	DB2
SNAP_GET_UTIL_PROGRESS	DB2

Procedure

1. Connect to a database. This can be any database in the instance you need to monitor. To be able to call a stored procedure, you must be connected to a database.
2. Determine the snapshot request type, and the database and partition you need to monitor.
3. Call the SNAP_WRITE_FILE stored procedure with the appropriate parameter settings for the snapshot request type, database, and partition. For example,

here is a call that will capture a snapshot of application information about the SAMPLE database for the current connected partition:

```
CALL SNAP_WRITE_FILE('APPL_ALL','SAMPLE',-1)
```

The SNAP_WRITE_FILE stored procedure has three input parameters:

- a snapshot request type (see Table 26 on page 259, which provides a cross-reference of the snapshot table functions and their corresponding request types)
- a VARCHAR (128) for the database name. If you enter NULL, the name of the currently connected database is used.

  Note: This parameter does not apply to the database manager level snapshot table functions; they only have parameters for request type and partition number.
- a SMALLINT for the partition number (a value between 0 and 999). For the partition number parameter, enter the integer corresponding to partition number you wish to monitor. To capture a snapshot for the currently connected partition, enter a value of -1 or a NULL. To capture a global snapshot, enter a value of -2.

Results

Once the snapshot data has been saved to a file, all users can issue queries with the corresponding snapshot table functions, specifying (NULL, NULL) as input values for database-level table functions, and (NULL) for database manager level table functions. The monitor data they receive is pulled from the files generated by the SNAP_WRITE_FILE stored procedure.

Note: While this provides a means to limit user access to sensitive monitor data, this approach does have some limitations:

- The snapshot monitor data available from the SNAP_WRITE_FILE files is only as recent as the last time the SNAP_WRITE_FILE stored procedure was called. You can ensure that recent snapshot monitor data is available by making calls to the SNAP_WRITE_FILE stored procedure at regular intervals. For instance, on UNIX systems you can set a cron job to do this.
- Users issuing queries with the snapshot table functions cannot identify a database or partition to monitor. The database name and partition number identified by the user issuing the SNAP_WRITE_FILE calls determine the contents of the files accessible by the snapshot table functions.
- If a user issues an SQL query containing a snapshot table function for which a corresponding SNAP_WRITE_FILE request type has not been run, a direct snapshot is attempted for the currently connected database and partition. This operation is successful only if the user has SYSADM, SYSCTRL, SYSMAINT, or SYSMON authority.

Accessing database system snapshots using snapshot table functions in SQL queries (with file access)

For every request type that authorized users have called the SNAP_WRITE_FILE stored procedure, any user can issue queries with the corresponding snapshot table functions. The monitor data they receive will be retrieved from the files generated by the SNAP_WRITE_FILE stored procedure.

Before you begin

For every snapshot table function with which you intend to access SNAP_WRITE_FILE files, an authorized user must have issued a SNAP_WRITE_FILE stored procedure call with the corresponding snapshot request types. If you issue an SQL query containing a snapshot table function for which a corresponding SNAP_WRITE_FILE request type has not been run, a direct snapshot is attempted for the currently connected database and partition. This operation is successful only if the user has SYSADM, SYSCTRL, SYSMAINT, or SYSMON authority.

About this task

Users who access snapshot data from SNAP_WRITE_FILE files with snapshot table functions cannot identify a database or partition to monitor. The database name and partition number identified by the user issuing the SNAP_WRITE_FILE calls determine the contents of the SNAP_WRITE_FILE files. The snapshot monitor data available from the SNAP_WRITE_FILE files is only as recent as the last time the SNAP_WRITE_FILE stored procedure captured snapshots.

Procedure

1. Connect to a database. This can be any database in the instance you need to monitor. To issue an SQL query with a snapshot table function, you must be connected to a database.
2. Determine the type of snapshot you need to capture.
3. Issue a query with the appropriate snapshot table function. For example, here is a query that will capture a snapshot of table space information:

   ```
   SELECT * FROM TABLE(SNAP_GET_TBSP_V91 (CAST(NULL AS VARCHAR(1)),
                       CAST (NULL AS INTEGER))) AS SNAP_GET_TBSP_V91
   ```

   Note: You must enter NULL values for the database name and partition number parameters. The database name and partition for the snapshot are determined in the call of the SNAP_WRITE_FILE stored procedure. Also, the database name parameter does not apply to the database manager level snapshot table functions; they only have a parameter for partition number. Each snapshot table function returns a table with one or more rows, with each column representing a monitor element. Accordingly, the monitor element column names correlate to the monitor element names.
4. You can also select individual monitor elements from the returned table. For example, the following statement will return only the **agent_id** monitor element:

   ```
   SELECT agent_id FROM TABLE(
                             SNAP_GET_APPL_V95(CAST(NULL AS VARCHAR(1)),
                                               CAST (NULL AS INTEGER)))
                             as SNAP_GET_APPL_V95
   ```

Snapshot monitor SQL Administrative Views

There are a number of different snapshot monitor SQL administrative views available, each returning monitor data about a specific area of the database system. For example, the SYSIBMADM.SNAPBP SQL administrative view captures a snapshot of buffer pool information. The following table lists each available snapshot monitor administrative view.

Table 27. Snapshot Monitor SQL Administrative Views

Monitor level	SQL Administrative Views	Information returned
Database manager	SYSIBMADM.SNAPDBM	Database manager level information.
Database manager	SYSIBMADM.SNAPFCM	Database manager level information regarding the fast communication manager (FCM).
Database manager	SYSIBMADM.SNAPFCM_PART	Database manager level information for a partition regarding the fast communication manager (FCM).
Database manager	SYSIBMADM.SNAPSWITCHES	Database manager monitor switch settings.
Database manager	SYSIBMADM.SNAPDBM_MEMORY_POOL	Database manager level information about memory usage.
Database	SYSIBMADM.SNAPDB	Database level information and counters for a database. Information is returned only if there is at least one application connected to the database.
Database	SYSIBMADM.SNAPDB_MEMORY_POOL	Database level information about memory usage for UNIX platforms only.
Database	SYSIBMADM.SNAPHADR	Database level information about high availability disaster recovery.
Application	SYSIBMADM.SNAPAPPL	General application level information for each application that is connected to the database. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	SYSIBMADM.SNAPAPPL_INFO	General application level identification information for each application that is connected to the database.
Application	SYSIBMADM.SNAPLOCKWAIT	Application level information regarding lock waits for the applications connected to the database.
Application	SYSIBMADM.SNAPSTMT	Application level information regarding statements for the applications connected to the database. This includes the most recent SQL statement executed (if the statement switch is set).
Application	SYSIBMADM.SNAPAGENT	Application level information regarding the agents associated with applications connected to the database.
Application	SYSIBMADM.SNAPSUBSECTION	Application level information regarding the subsections of access plans for the applications connected to the database.
Application	SYSIBMADM.SNAPAGENT_MEMORY_POOL	Information about memory usage at the agent level.
Table	SYSIBMADM.SNAPTAB	Table activity information at the database and application level for each application connected to the database. Table activity information at the table level for each table that *was accessed* by an application connected to the database. Requires the table switch.

Table 27. Snapshot Monitor SQL Administrative Views (continued)

Monitor level	SQL Administrative Views	Information returned
Table	SYSIBMADM.SNAPTAB_REORG	Table reorganization information at the table level for each table in the database undergoing reorganization.
Lock	SYSIBMADM.SNAPLOCK	Lock information at the database level, and application level for each application connected to the database. Requires the lock switch.
Table space	SYSIBMADM.SNAPTBSP	Information about table space activity at the database level, the application level for each application connected to the database, and the table space level for each table space that has been accessed by an application connected to the database. Requires the buffer pool switch.
Table space	SYSIBMADM.SNAPTBSP_PART	Information about table space configuration.
Table space	SYSIBMADM.SNAPTBSP_QUIESCER	Information about quiescers at the table space level.
Table space	SYSIBMADM.SNAPCONTAINER	Information about table space container configuration at the table space level.
Table space	SYSIBMADM.SNAPTBSP_RANGE	Information about ranges for a table space map.
Buffer pool	SYSIBMADM.SNAPBP	Buffer pool activity counters for the specified database. Requires the buffer pool switch.
Buffer pool	SYSIBMADM.SNAPBP_PART	Information on buffer size and usage, calculated per partition.
Dynamic SQL	SYSIBMADM.SNAPDYN_SQL	Point–in–time statement information from the SQL statement cache for the database.
Database	SYSIBMADM.SNAPUTIL	Information about utilities.
Database	SYSIBMADM.SNAPUTIL_PROGRESS	Information about the progress of utilities.
Database	SYSIBMADM.SNAPDETAILLOG	Database level information about log files.
Database	SYSIBMADM.SNAPSTORAGE_PATHS	Returns a list of automatic storage paths for the database including file system information for each storage path.

Before capturing a snapshot, consider if you need information from monitor elements that are under monitor switch control. If a particular monitor switch is off, the monitor elements under its control will not be collected. See the individual monitor elements to determine if an element you need is under switch control.

All snapshot monitoring administrative views and associated table functions use a separate instance connection, which is different from the connection the current session uses. Therefore, an implicit instance attachment can be established, and only default database manager monitor switches are effective. Ineffective monitor switches include any that are turned on or off dynamically from the current session or application.

DB2 Version 9.5 also provides you with a set of administrative views that do not only return values of individual monitor elements, but also return computed values that are commonly required in monitoring tasks. For example, the

SYSIBMADM.BP_HITRATIO administrative view returns calculated values for buffer pool hit ratios, which combine a number of individual monitor elements.

Table 28. Snapshot Monitor SQL Administrative Convenience Views

SQL Administrative Convenience Views	Information returned
SYSIBMADM.APPLICATIONS	Information about connected database applications.
SYSIBMADM.APPL_PERFORMANCE	Information about the rate of rows selected versus the number of rows read by an application.
SYSIBMADM.BP_HITRATIO	Buffer pool hit ratios, including total, data, and index, in the database.
SYSIBMADM.BP_READ_IO	Information about buffer pool read performance.
SYSIBMADM.BP_WRITE_IO	Information about buffer pool write performance.
SYSIBMADM.CONTAINER_UTILIZATION	Information about table space containers and utilization rates.
SYSIBMADM.LOCKS_HELD	Information on current locks held.
SYSIBMADM.LOCKWAITS	Information about DB2 agents working on behalf of applications that are waiting to obtain locks.
SYSIBMADM.LOG_UTILIZATION	Information about log utilization for the currently connected database.
SYSIBMADM.LONG_RUNNING_SQL	Information about the longest running SQL in the currently connected database.
SYSIBMADM.QUERY_PREP_COST	Information about the time required to prepare different SQL statements.
SYSIBMADM.TBSP_UTILIZATION	Table space configuration and utilization information.
SYSIBMADM.TOP_DYNAMIC_SQL	The top dynamic SQL statements sortable by number of executions, average execution time, number of sorts, or sorts per statement.

SQL access to database system snapshots

About this task

There are two ways to access snapshot monitor data with the snapshot monitor SQL table functions (referred to as *snapshot table functions*):

- direct access
- file access

Direct access

Authorized users can issue queries with snapshot table functions and receive result sets containing monitor data. With this approach, access to snapshot monitor data is only available to users that have SYSADM, SYSCTRL, SYSMAINT, or SYSMON authority.

To capture snapshot information using direct access:

1. Optional: Set and check the status of the monitor switches.
2. Capture database system snapshots using SQL.

File access

Authorized users call the SNAPSHOT_FILEW stored procedure, identifying the snapshot request type, and the affected partition and database. The SNAPSHOT_FILEW stored procedure then saves the monitor data into a file on the database server.

Every request type for which authorized users can call the SNAPSHOT_FILEW stored procedure,

While this is a safe means of providing all users with access to snapshot monitor data, there are limitations to this approach:

- The snapshot monitor data available from the SNAPSHOT_FILEW files is only as recent as the last time the SNAPSHOT_FILEW stored procedure was called. You can ensure that recent snapshot monitor data is available by making calls to the SNAPSHOT_FILEW stored procedure at regular intervals. For instance, on UNIX systems you can set a cron job to do this.
- Users issuing queries with the snapshot table functions cannot identify a database or partition to monitor. The database name and partition number identified by the user issuing the SNAPSHOT_FILEW calls determine the contents of the files accessible by the snapshot table functions.
- If a user issues an SQL query containing a snapshot table function for which a corresponding SNAPSHOT_FILEW request type has not been run, a direct snapshot is attempted for the currently connected database and partition. This operation is successful only if the user has SYSADM, SYSCTRL, SYSMAINT, or SYSMON authority.

The following tasks are performed by the SYSADM, SYSCTRL, SYSMAINT, or SYSMON user who captures database system snapshot information to a file.

Procedure

1. Find out the needs of users who will issue snapshot requests. Specifically, determine the monitor data they need, the database it is to be collected from, and if the collection needs to be limited to a particular partition.
2. Optional: Set and check the status of the monitor switches.
3. Capture database system snapshot information to a file .

What to do next

Once the SYSADM, SYSCTRL, SYSMAINT, or SYSMON user has completed the preceding steps, all users can access database system snapshot information using snapshot table functions in SQL queries.

Capturing a database snapshot from the CLP

You can capture database snapshots from the CLP using the GET SNAPSHOT command. A number of different snapshot request types are available, which can be accessed by specifying certain parameters for the GET SNAPSHOT command.

Before you begin

You must have SYSADM, SYSCTRL, SYSMAINT, or SYSMON authority to capture a database snapshot.

You must have an instance attachment to capture a database snapshot. If there is not an attachment to an instance, then a default instance attachment is created. To obtain a snapshot of a remote instance, you must first attach to that instance.

Procedure

- Optional: Set and check the status of the monitor switches.
- From the CLP, issue the `GET SNAPSHOT` command with the desired parameters. In the following example, the snapshot captures information for all databases:

```
db2 get snapshot for all databases
```

To capture a database snapshot for a specific database, use the following command:

```
db2 get snapsot for database on db-name
```

where *db-name* is the name of the database you are interested in.

- The following example captures database manager level information:

  ```
  db2 get snapshot for dbm
  ```

- For partitioned database systems, you can capture a database snapshot specifically for a certain partition, or globally for all partitions. To capture a database snapshot for all applications on a specific partition (for example, partition number 2), issue the following command:

  ```
  db2 get snapshot for all applications at dbpartitionnum 2
  ```

- To capture a database snapshot for all applications on all partitions, issue the following command:

  ```
  db2 get snapshot for all applications global
  ```

  For global snapshots on partitioned databases, the monitor data from all the partitions is aggregated.

Snapshot monitor CLP commands

The following table lists all the supported snapshot request types. For certain request types, some information is returned only if the associated monitor switch is set ON. See the individual monitor elements to determine if a required element is under switch control.

Table 29. Snapshot Monitor CLP Commands

Monitor level	CLP command	Information returned
Connections list	`list applications [show detail]`	Application identification information for all applications currently connected to a database that is managed by the DB2 instance on the partition where snapshot is taken.
Connections list	`list applications for database dbname [show detail]`	Application identification information for each application currently connected to the specified database.
Connections list	`list dcs applications`	Application identification information for all DCS applications currently connected to a database that is managed by the DB2 instance on the partition where snapshot is taken.
Database manager	`get snapshot for dbm`	Database manager level information, including instance-level monitor switch settings.
Database manager	`get dbm monitor switches`	Instance-level monitor switch settings.
Database	`get snapshot for database on dbname`	Database level information and counters for a database. Information is returned only if there is at least one application connected to the database.
Database	`get snapshot for all databases`	Database level information and counters for each database active on the partition. Information is returned only if there is at least one application connected to the database.

Table 29. Snapshot Monitor CLP Commands (continued)

Monitor level	CLP command	Information returned
Database	`list active databases`	The number of connections to each active database. Includes databases that were started using the ACTIVATE DATABASE command, but have no connections.
Database	`get snapshot for dcs database on dbname`	Database level information and counters for a specific DCS database. Information is returned only if there is at least one application connected to the database.
Database	`get snapshot for remote database on dbname`	Database level information and counters for a specific federated system database. Information is returned only if there is at least one application connected to the database.
Database	`get snapshot for all remote databases`	Database level information and counters for each active federated system database on the partition. Information is returned only if there is at least one application connected to the database.
Application	`get snapshot for application applid appl-id`	Application level information, including cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	`get snapshot for application agentid appl-handle`	Application level information, includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	`get snapshot for applications on dbname`	Application level information for each application that is connected to the database on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	`get snapshot for all applications`	Application level information for each application that is active on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	`get snapshot for dcs application applid appl-id`	Application level information, including cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	`get snapshot for all dcs applications`	Application level information for each DCS application that is active on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	`get snapshot for dcs application agentid appl-handle`	Application level information, including cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	`get snapshot for dcs applications on dbname`	Application level information for each DCS application that is connected to the database on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	`get snapshot for remote applications on dbname`	Application level information, includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).

Table 29. Snapshot Monitor CLP Commands (continued)

Monitor level	CLP command	Information returned
Application	`get snapshot for all remote applications`	Application level information for each federated system application that is active on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Table	`get snapshot for tables on dbname`	Table activity information at the database and application level for each application connected to the database. Table activity information at the table level for each table that **was accessed** by an application connected to the database. Requires the table switch.
Lock	`get snapshot for locks for application applid appl-id`	List of locks held by the application. Lock wait information requires the lock switch.
Lock	`get snapshot for locks for application agentid appl-handle`	List of locks held by the application. Lock wait information requires the lock switch.
Lock	`get snapshot for locks on dbname`	Lock information at the database level, and application level for each application connected to the database. Requires the lock switch.
Table space	`get snapshot for tablespaces on dbname`	Information about table space activity for a database. Requires the buffer pool switch. Also included is information on containers, quiescers, and ranges. This information is not under switch control.
Buffer pool	`get snapshot for all bufferpools`	Buffer pool activity counters. Requires the buffer pool switch.
Buffer pool	`get snapshot for bufferpools on dbname`	Buffer pool activity counters for the specified database. Requires the buffer pool switch.
Dynamic SQL	`get snapshot for dynamic sql on dbname`	Point-in-time statement information from the SQL statement cache for the database. The information can also be from a remote data source.

Capturing a database snapshot from a client application

You can capture database snapshots using the snapshot monitor API in a C, C++, or a COBOL application. In C and C++ a number of different snapshot request types can be accessed by specifying certain parameters in db2GetSnapshot().

Before you begin

You must have SYSADM, SYSCTRL, SYSMAINT, or SYSMON authority to use the db2MonitorSwitches API.

You must have an instance attachment to capture a database snapshot. If there is not an attachment to an instance, then a default instance attachment is created. To obtain a snapshot of a remote instance, you must first attach to that instance.

Procedure

1. Optional: Set and check the status of the monitor switches.
2. Include the following DB2 libraries: sqlmon.h and db2ApiDf.h. These are found in the `include` subdirectory under `sqllib`.

```
#include <db2ApiDf.h>
#include <sqlmon.h>
```

3. Set snapshot buffer unit size to 100 KB.

```
#define SNAPSHOT_BUFFER_UNIT_SZ 102400
```

4. Declare the sqlca, sqlma, db2GetSnapshotData, and sqlm_collected structures. Also, initialize a pointer to contain the snapshot buffer, and establish the buffer's size.

```
struct sqlma *pRequestedDataGroups;
struct sqlca sqlca;
memset (&sqlca, '\0', sizeof(struct sqlca));
struct sqlm_collected collectedData;
memset (&collectedData, '\0', sizeof(collectedData));
db2GetSnapshotData getSnapshotParam;
memset (&getSnapshotParam, '\0', sizeof(getSnapshotParam));

static sqluint32 snapshotBufferSize = SNAPSHOT_BUFFER_UNIT_SZ;
sqluint32 outputFormat;
char *snapshotBuffer;
```

5. Initialize the sqlma structure and specify that the snapshot to be captured is of database manager level information.

```
pRequestedDataGroups = (struct sqlma *)malloc(SQLMASIZE(1));
memset(pRequestedDataGroups, '\0', SQLMASIZE(1));
pRequestedDataGroups->obj_num = 1;
pRequestedDataGroups->obj_var[0].obj_type = SQLMA_DB2;
```

6. Initialize the buffer which is to hold the snapshot output.

```
snapshotBuffer = (char *)malloc(snapshotBufferSize);
memset (snapshotBuffer, '\0', snapshotBufferSize);
```

7. Populate the db2GetSnapshotData structure with the snapshot request type (from the sqlma structure), buffer information, and other information required to capture a snapshot.

```
getSnapshotParam.piSqlmaData = pRequestedDataGroups;
getSnapshotParam.poCollectedData = &collectedData;
getSnapshotParam.poBuffer = snapshotBuffer;
getSnapshotParam.iVersion = SQLM_DBMON_VERSION9_5;
getSnapshotParam.iBufferSize = snapshotBufferSize;
getSnapshotParam.iStoreResult = 0;
getSnapshotParam.iNodeNumber = SQLM_CURRENT_NODE;
getSnapshotParam.poOutputFormat = &outputFormat;
getSnapshotParam.iSnapshotClass = SQLM_CLASS_DEFAULT;
```

8. Capture the snapshot. Pass the db2GetSnapshotData structure, which contains the information necessary to capture a snapshot, as well as a reference to the buffer, where snapshot output is to be directed.

```
db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
```

9. Include logic to handle buffer overflow. After a snapshot is taken, the sqlcode is checked for a buffer overflow. If a buffer overflow occurred the buffer is cleared and reinitialized, and the snapshot is taken again.

```
while (sqlca.sqlcode == 1606)
{
  free(snapshotBuffer);
  snapshotBufferSize = snapshotBufferSize +
  SNAPSHOT_BUFFER_UNIT_SZ;
  snapshotBuffer = (char *)malloc(snapshotBufferSize);
  if (snapshotBuffer == NULL)
  {
    printf("\nMemory allocation error.\n");
    return 1;
  }
  getSnapshotParam.iBufferSize = snapshotBufferSize;
  getSnapshotParam.poBuffer = snapshotBuffer;
  db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
}
```

10. Process the snapshot monitor data stream.
11. Clear the buffer.

```
free(snapshotBuffer);
free(pRequestedDataGroups);
```

Snapshot monitor API request types

The following table lists all the supported snapshot request types. For certain request types, some information is returned only if the associated monitor switch is set ON. See the individual monitor elements to determine if a required element is under switch control.

Table 30. Snapshot Monitor API Request Types

Monitor level	API request type	Information returned
Connections list	SQLMA_APPLINFO_ALL	Application identification information for all applications currently connected to a database that is managed by the DB2 instance on the partition where snapshot is taken.
Connections list	SQLMA_DBASE_APPLINFO	Application identification information for each application currently connected to the specified database.
Connections list	SQLMA_DCS_APPLINFO_ALL	Application identification information for all DCS applications currently connected to a database that is managed by the DB2 instance on the partition where snapshot is taken.
Database manager	SQLMA_DB2	Database manager level information, including instance-level monitor switch settings.
Database	SQLMA_DBASE	Database level information and counters for a database. Information is returned only if there is at least one application connected to the database.
Database	SQLMA_DBASE_ALL	Database level information and counters for each database active on the partition. The number of connections to each active database. Includes databases that were started using the ACTIVATE DATABASE command, but have no connections. Information is returned only if there is at least one application connected to the database.
Database	SQLMA_DCS_DBASE	Database level information and counters for a specific DCS database. Information is returned only if there is at least one application connected to the database.
Database	SQLMA_DCS_DBASE_ALL	Database level information and counters for each DCS database active on the partition. Information is returned only if there is at least one application connected to the database.
Database	SQLMA_DBASE_REMOTE	Database level information and counters for a specific federated system database. Information is returned only if there is at least one application connected to the database.
Database	SQLMA_DBASE_REMOTE_ALL	Database level information and counters for each active federated system database on the partition. Information is returned only if there is at least one application connected to the database.

Table 30. Snapshot Monitor API Request Types (continued)

Monitor level	API request type	Information returned
Application	SQLMA_APPL	Application level information, including cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	SQLMA_AGENT_ID	Application level information, includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	SQLMA_DBASE_APPLS	Application level information for each application that is connected to the database on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	SQLMA_APPL_ALL	Application level information for each application that is active on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	SQLMA_DCS_APPL	Application level information, including cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	SQLMA_DCS_APPL_ALL	Application level information for each DCS application that is active on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	SQLMA_DCS_APPL_HANDLE	Application level information, including cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	SQLMA_DCS_DBASE_APPLS	Application level information for each DCS application that is connected to the database on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	SQLMA_DBASE_APPLS_REMOTE	Application level information, includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	SQLMA_APPL_REMOTE_ALL	Application level information for each federated system application that is active on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Table	SQLMA_DBASE_TABLES	Table activity information at the database and application level for each application connected to the database. Table activity information at the table level for each table that **was accessed** by an application connected to the database. Requires the table switch.
Lock	SQLMA_APPL_LOCKS	List of locks held by the application. Lock wait information requires the lock switch.

Table 30. Snapshot Monitor API Request Types (continued)

Monitor level	API request type	Information returned
Lock	SQLMA_APPL_LOCKS_AGENT_ID	List of locks held by the application. Lock wait information requires the lock switch.
Lock	SQLMA_DBASE_LOCKS	Lock information at the database level, and application level for each application connected to the database. Requires the lock switch.
Table space	SQLMA_DBASE_TABLESPACES	Information about table space activity at the database level, the application level for each application connected to the database, and the table space level for each table space that has been accessed by an application connected to the database. Requires the buffer pool switch.
Buffer pool	SQLMA_BUFFERPOOLS_ALL	Buffer pool activity counters. Requires the buffer pool switch.
Buffer pool	SQLMA_DBASE_BUFFERPOOLS	Buffer pool activity counters for the specified database. Requires the buffer pool switch.
Dynamic SQL	SQLMA_DYNAMIC_SQL	Point-in-time statement information from the SQL statement cache for the database.

Snapshot monitor sample output

To illustrate the nature of the snapshot monitor, here is an example of a snapshot being taken using the CLP, along with its corresponding output. The objective in this example is to obtain a list of the locks held by applications connected to the SAMPLE database. The steps taken are as follows:

1. Connect to the sample database:

   ```
   db2 connect to sample
   ```

2. Turn on the LOCK switch with the UPDATE MONITOR SWITCHES command, so that the time spent waiting for locks is collected:

   ```
   db2 update monitor switches using LOCK on
   ```

3. Issue a command or statement that will require locks on the database catalogs. In this case, we will declare, open, and fetch a cursor:

   ```
   db2 -c- declare c1 cursor for
                         select * from staff where job='Sales' for update
   db2 -c- open c1
   db2 -c- fetch c1
   ```

4. Take the database lock snapshot, using the GET SNAPSHOT command:

   ```
   db2 get snapshot for locks on sample
   ```

After the GET SNAPSHOT command is issued from the CLP, the snapshot output is directed to the screen.

```
            Database Lock Snapshot

Database name                               = SAMPLE
Database path                               = C:\DB2\NODE0000\SQL00001\
Input database alias                        = SAMPLE
Locks held                                  = 5
Applications currently connected            = 1
Agents currently waiting on locks           = 0
Snapshot timestamp                          = 06-05-2002 17:08:25.048027

Application handle                          = 8
Application ID                              = *LOCAL.DB2.0098C5210749
```

```
Sequence number                          = 0001
Application name                         = db2bp.exe
CONNECT Authorization ID                 = DB2ADMIN
Application status                       = UOW Waiting
Status change time                       = Not Collected
Application code page                    = 1252
Locks held                               = 5
Total wait time (ms)                     = 0

List Of Locks
 Lock Name                  = 0x02000300050000000000000052
 Lock Attributes            = 0x00000000
 Release Flags              = 0x00000001
 Lock Count                 = 1
 Hold Count                 = 0
 Lock Object Name           = 5
 Object Type                = Row
 Tablespace Name            = USERSPACE1
 Table Schema               = DB2ADMIN
 Table Name                 = STAFF
 Mode                       = U

 Lock Name                  = 0x02000300000000000000000054
 Lock Attributes            = 0x00000000
 Release Flags              = 0x00000001
 Lock Count                 = 1
 Hold Count                 = 0
 Lock Object Name           = 3
 Object Type                = Table
 Tablespace Name            = USERSPACE1
 Table Schema               = DB2ADMIN
 Table Name                 = STAFF
 Mode                       = IX

 Lock Name                  = 0x0100000001000000100810056
 Lock Attributes            = 0x00000000
 Release Flags              = 0x40000000
 Lock Count                 = 1
 Hold Count                 = 0
 Lock Object Name           = 0
 Object Type                = Internal Variation Lock
 Mode                       = S

 Lock Name                  = 0x41414141414A48520000000041
 Lock Attributes            = 0x00000000
 Release Flags              = 0x40000000
 Lock Count                 = 1
 Hold Count                 = 0
 Lock Object Name           = 0
 Object Type                = Internal Plan Lock
 Mode                       = S

 Lock Name                  = 0x434F4E544F4B4E310000000041
 Lock Attributes            = 0x00000000
 Release Flags              = 0x40000000
 Lock Count                 = 1
 Hold Count                 = 0
 Lock Object Name           = 0
 Object Type                = Internal Plan Lock
 Mode                       = S
```

From this snapshot, you can see that there is currently one application connected to the SAMPLE database, and it is holding five locks.

```
Locks held                               = 5
Applications currently connected         = 1
```

Note that the time (`Status change time`) when the `Application status` became `UOW Waiting` is returned as `Not Collected`. This is because the UOW switch is OFF.

The lock snapshot also returns the total time spent so far in waiting for locks, by applications connected to this database.

```
Total wait time (ms)                     = 0
```

Subsection snapshots

On systems that use inter-partition parallelism, the SQL compiler partitions the access plan for an SQL statement into subsections. Each subsection is executed by a different DB2 agent (or agents for SMP).

The access plan for an SQL statement generated by the DB2 code generator during compilation can be obtained using the db2expln command. As an example, selecting all the rows from a table that is partitioned across several partitions might result in an access plan having two subsections:

1. Subsection 0, the coordinator subsection, whose role is to collect rows fetched by the other DB2 agents (subagents) and return them to the application.
2. Subsection 1, whose role is to perform a table scan and return the rows to the coordinating agent.

In this simple example, subsection 1 would be distributed across all the database partitions. There would be a subagent executing this subsection on each physical partition of the database partition group to which this table belongs.

The database system monitor allows you to correlate run-time information with the access plan, which is compile-time information. With inter-partition parallelism, the monitor breaks information down to the subsection level. For example, when the statement monitor switch is ON, a GET SNAPSHOT FOR APPLICATION will return information for each subsection executing on this partition, as well as totals for the statement.

The subsection information returned for an application snapshot includes:

- the number of table rows read/written
- CPU consumption
- elapsed time
- the number of table queue rows sent and received from other agents working on this statement. This allows you to track the execution of a long running query by taking a series of snapshots.
- subsection status. If the subsection is in a WAIT state, because it is waiting for another agent to send or receive data, then the information also identifies the partition or partitions preventing the subsection from progressing in its execution. You may then take a snapshot on these partitions to investigate the situation.

The information logged by a statement event monitor for each subsection after it has finished executing includes: CPU consumption, total execution, time, and several other counters.

Global snapshots on partitioned database systems

On a partitioned database system, you can use the snapshot monitor to take a snapshot of the current partition, a specified partition, or all partitions. When taking a global snapshot across all the partitions of a partitioned database, data is aggregated before the results are returned.

Data is aggregated for the different element types as follows:

- **Counters, Time, and Gauges**

  Contains the sum of all like values collected from each partition in the instance. For example, GET SNAPSHOT FOR DATABASE XYZ ON TEST GLOBAL would return the number of rows read (rows_read) from the database for all partitions in the partitioned database instance.
- **Watermarks**

  Returns the highest (for high water) or lowest (for low water) value found for any partition in the partitioned database system. If the value returned is of concern, then snapshots for individual partitions can be taken to determine if a particular partition is over utilized, or if the problem is instance-wide.
- **Timestamp**

  Set to the timestamp value for the partition where the snapshot monitor instance agent is attached. Note that all timestamp values are under control of the `timestamp` monitor switch.
- **Information**

  Returns the most significant information for a partition that may be impeding work. For example, for the element `appl_status`, if the status on one partition was UOW Executing, and on another partition Lock Wait, Lock Wait would be returned, since it is the state that's holding up execution of the application.

You can also reset counters, set monitor switches, and retrieve monitor switch settings for individual partitions or all partitions in your partitioned database.

Note: When taking a global snapshot, if one or more partitions encounter an error, then data is collected from the partitions where the snapshot was successful and a warning (sqlcode 1629) is also returned. If a global get or update of monitor switches, or a counter reset fails on one or more partitions, then those partitions will not have their switches set, or data reset.

Snapshot monitor self-describing data stream

After you capture a snapshot with the db2GetSnapshot API, the API returns the snapshot output as a self-describing data stream. Figure 13 on page 276 shows the structure of the data stream and Table 31 on page 276 provides some examples of the logical data groups and monitor elements that might be returned.

Note: Descriptive names are used for the identifiers in the examples and tables. These names are prefixed by **SQLM_ELM_** in the actual data stream. For example, collected would appear as SQLM_ELM_COLLECTED in the snapshot monitor output. Types are prefixed with **SQLM_TYPE_** in the actual data stream. For example, headers appear as SQLM_TYPE_HEADER in the data stream.

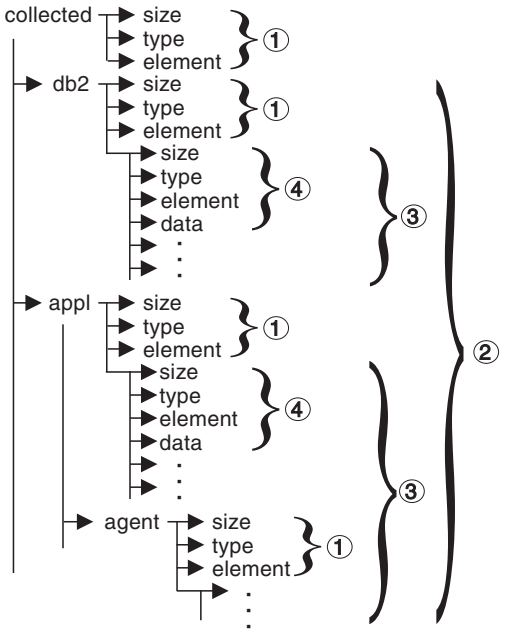


Figure 13. Snapshot Monitor Data Stream

1. Each logical data group begins with a header that indicates its size and name. This size does not include the volume of data taken up by the header itself.
2. Size in the collected header returns the total size of the snapshot.
3. The size element in other headers indicates the size of all the data in that logical data group, including any subordinate groupings.
4. Monitor element information follows its logical data group header and is also self-describing.

Table 31. Sample Snapshot Data Stream

Logical Data Group	Data Stream	Description
collected		
	1000	Size of snapshot data (in bytes).
	header	Indicates the start of a logical data group.
	collected	Name of the logical data group.
	4	Size of the data stored in this monitor element.
	u32bit	Monitor element type - unsigned 32 bit numeric.
	server_db2_type	The name of the monitor element collected.
	sqlf_nt_server	The collected value for this element.
	2	Size of the data stored in this monitor element.
	u16bit	Monitor element type - unsigned 16 bit numeric.
	node_number	The name of the monitor element collected.
	3	The collected value for this element.

Table 31. Sample Snapshot Data Stream (continued)

Logical Data Group	Data Stream	Description
db2		
	200	Size of the DB2 level portion of data in the snapshot.
	header	Indicates the start of a logical data group.
	db2	Name of the logical data group.
	4	Size of the data stored in this monitor element.
	u32bit	Monitor element type - unsigned 32 bit numeric.
	sort_heap_allocated	The name of the monitor element collected.
	16	The collected value for this element.
	4	Size of the data stored in this monitor element.
	u32bit	Monitor element type - unsigned 32 bit numeric.
	local_cons	The name of the monitor element collected.
	3	The collected value for this element.
	. . .	. . .
appl		
	100	Size of the appl element data in the snapshot.
	header	Indicates the start of a logical data group.
	appl	Name of the logical data group.
	4	Size of the data stored in this monitor element.
	u32bit	Monitor element type - unsigned 32 bit numeric.
	locks_held	The name of the monitor element collected.
	3	The collected value for this element.
	. . .	. . .
agent		
	50	Size of the agent portion of the appl structure.
	header	Indicates the start of a logical data group.
	agent	Name of the logical data group.
	4	Size of the data stored in this monitor element.
	u32bit	Monitor element type - 32 bit numeric.
	agent_pid	The name of the monitor element collected.
	12	The collected value for this element.
	. . .	. . .

The db2GetSnapshot() routine returns the self-describing snapshot data in the user-supplied buffer. Data is returned in the logical data groupings associated with the type of snapshot being captured.

Each item returned by a snapshot request contains fields that specify its size and type. The size can be used to parse through the returned data. A field's size can also be used to skip over a logical data group. For example, to skip over the DB2 record you need to determine the number of bytes in the data stream. Use the following formula to calculate the number of bytes to skip:

```
size of the db2 logical data grouping + sizeof(sqlm_header_info)
```

Monitoring with db2top in interactive mode commands

The **db2top** monitoring utility quickly and efficiently monitors a complex DB2 environment. It combines DB2 snapshot information from all database partitions and provides a dynamic, real-time view of a running DB2 system using a text-based user interface.

About this task

When you run **db2top** in interactive mode, you can issue the following commands:

A Monitor either the primary or the secondary database in a HADR cluster.

a Goto application details for agent (or restrict on agent on statement screen). The **db2top** command will prompt for the agent-id.

B Display the main consumer of critical server resources (Bottleneck Analysis).

c This option allows to change the order of the columns displayed on the screen. The syntax is in the form: 1,2,3,... where 1,2,3 correspond respectively to the 1st, 2nd and 3rd columns displayed. These are the column numbers to use when specifying a sort criteria.

When you use the `c` switch key, a screen is shown specifying the order of the columns displayed on screen. The left part of the screen displays the default order and column numbers; the right part of the screen displays the current ordering. To change the order of the columns, enter the new column order in the text field at the bottom of the screen. Next, enter the relative column positions as displayed on the left, separated by commas. Not all columns need to be specified. This column ordering can be saved in *$DB2TOPRC* for subsequent **db2top** monitoring sessions by selecting w. You can sort and select in which order the columns are displayed on the screen. Valid keywords for column ordering in the `.db2toprc` file are:

- sessions=
- tables=
- tablespaces=
- bufferpools=
- dynsql=
- statements=
- locks=
- utilities=
- federation=

b Goto buffer pool screen.

C Toggle snapshot data collector on/off.

d Goto database screen.

D Goto the dynamic SQL screen.

f Freeze screen.

F Monitor federated queries on the primary server.

G Toggle graph on/off.

h Go to Help screen

H Goto the history screen

i Toggle idle sessions on/off.

k Toggle actual vs delta values.

l Goto sessions screen.

L Allows to display the complete query text from the SQL screen. Regular DB2 explain can then be run using e or X options.

m Display memory pools.

o Display session setup.

p Goto the partitions screen.

P Select db partition on which to issue snapshot.

q Quit db2top.

R Reset snapshot data.

s Goto the statements screen.

S Run native DB2 snapshot.

t Goto table spaces screen.

T Goto tables screen

u Display active utilities and aggregate them across database partitions.

U Goto the locks screen.

V Set default explains schema.

w Write session settings to .db2toprc.

W Watch mode for agent_id, os_user, db_user, application or netname. Statements returned by the session snapshot (option l) will be written to agent.sql, os_user-agent.sql, db_user-agent.sql, application- agent.sql or netname-agent.sql. When issued from the dynamic SQL screen (option D), statements will be written to db2adv.sql in a format compatible with db2advis.

X Toggle extended mode on/off.

z | Z Sort on ascending or descending order.

/ Enter expression to filter data. Expression must conform to regular expression. Each function (screen) can be filtered differently. The regexp check is applied to the whole row.

< | > Move to left or right of screen.

The following switches apply to the applications screen only:

r Return to previous function.

R Toggle automatic refresh.

g Toggle graph on/off.

X Toggle extended mode on/off.

d Display agents.

To start **db2top in interactive mode**, issue the following command:

```
db2top -d <database name>
```

When you type

```
db2top -d sample
```

the following output is displayed:

```
[\]11:57:10,refresh=2secs(0.000) Inactive,part=[1/1],<instanceName>:sample
[d=Y,a=N,e=N,p=ALL] [qp=off]

[/]: When rotating, it means that db2top is waiting between two snapshots, otherwise, it means db2top is waiting
     from an answer from DB2
11:57:10: current time
refresh=2secs: time interval
refresh=!secs: Exclamation mark means the time to process the snapshot by DB2 is longer than the refresh interval.
     In this case, db2top will increase the interval by 50%. If this occurs too often because the system is too busy,
     you can either increase the snapshot interval (option I), monitor a single database partition (option P), or turn
     off extended display mode (option x)
0.000 : time spent inside DB2 to process the snapshot
d=Y/N : delta or cumulative snapshot indicator (command option -k or option k).
a=Y/N : Active only or all objects indicator (-a command option set or i)
e=Y/N : Extended display indicator
p=ALL : All database partitions
p=CUR: Current database partition (-P command option with no partition number specified)
p=3 : target database partition number: say 3

Inactive: : Shows inactive if DB2 is not running, otherwise displays the platform on which DB2 is running
part=[1/1] : active database partition number vs total database partition number. For example, part=[2,3] means one
      database partition out of 3 is down (2 active, 3 total)
<instanceName> : instance name
sample : database name
qp=off/on : query patroller indicator (DYNMGMT database configuration parameter) for the database partition
      on which db2top is attached
```

Example

The following example demonstrates running the **db2top** monitoring utility in interactive mode in a partitioned database environment:

```
db2top -d TEST -n mynode -u user -p passwd -V skm4 -B -i 1
The command parameters are as follows:
-d TEST     # database name
-n mynode   #  node name
-u user     #  user id
-p passwd   #  password
-V skm4     #  Schema name
-B          #  Bold enabled
-i 1        #  Screen update interval: 1 second
```

.db2toprc configuration file

The `.db2toprc` configuration file is a user generated file used to set parameters at initialization time for the **db2top** monitoring utility.

The **db2top** utility will search for the location of the `.db2toprc` file using the user-defined variable *$db2topRC*. If the variable is not set, **db2top** will first search for the `.db2toprc` file in the current directory and then in the `home` directory. The `.db2toprc` file is user generated.

Environment variables

You can set the following environment variables:

- **DB2TOPRC**

  A user defined environment variable that stores the location of the `.db2toprc` file. For example, on Linux, you can define **DB2TOPRC** as: `export db2topRC="~/db2top"`.

  If the variable is not set by the user, **db2top** will first search for the `.db2toprc` file in the current directory and then in the `home` directory.

- **DB2DBDFT**

  This variable specifies the database alias name of the database to be used for implicit connects. It is used when no database name is specified on the command line or in the `.db2toprc` configuration file.
- **EDITOR**

  This system environment variable specifies the command used to start the text editor used to display the results of explain or native snapshots.

  If this variable is not set, **vi** is used.

Structure

Some of the entries in the `.db2toprc` file are described here.

cpu=command
: Use this entry to display the results of CPU activity on the second line at the right of the screen output. For example:

  ```
  cpu=vmstat 2 2 | tail -1 | awk '{printf("%d(usr+sys)",$14+$15);}'
  ```

  displays `Cpu=2(usr+sys)` on the right of the screen.

io=command
: Use this entry to specify a command and display the result on the second line at the left of the screen output. For example:

  ```
  io=vmstat 2 2 | tail -1 | awk '{printf("%d(bi+bo)",$10+$11);}'
  ```

  displays `Disk=76(bi+bo)` on the left of the screen.

  Both commands run as background processes and the fields on the screen are updated asynchronously.

shell alias=command
: Use this shell entry to specify a user defined command, for example: `shell M=top` spawns top from a **db2top** session when entering `M`. It returns to the current screen upon exit.

function alias=command
: Use this entry to specify a user defined command, for example: `function N=netstat` creates a new function called N that repeatedly displays the output of **netstat**. There can be multiple `function` entries. They must be placed on separate lines. For example:

  ```
  function Q=netstat
  function N=df -k
  ```

sort=command
: Use this entry to specify a sort order, for example: `sort=command` creates a default sort order for this function, where command is the column number. It can be either ascending or descending. Sort is valid for sessions, tables, tablespace, bufferpool, dynsql, statements, locks, utilities and federation.

Sample .db2toprc file

There is no default `.db2toprc` configuration file. However, you can press "W" to create a `.db2toprc` for the current setup. Use the following sample **.db2toprc** file as a reference. Comments have been added to all entries.

```
# db2top configuration file
# On UNIX, should be located in $HOME/.db2toprc
# File generated by db2top-1.0a
#
node= # [-n] nodename
database=sample # [-d] databasename
```

```
user= # [-u] database user
password= # [-p] user password (crypted)
schema= # [-V] default schema for explains
interval=2 # [-i] sampling interval
active=OFF # [-a] display active sessions only (on/off)
reset=OFF # [-R] Reset snapshot at startup (on/off)
delta=ON # [-k] Toggle display of delta/cumulative values (on/off)
gauge=ON # display graph on sessions list (on/off)
colors=ON
# True if terminal supports colors. Informs GE_WRS if it can display information with colors
graphic=ON  # True if terminal supports semi graphical characters (on/off).
port= # Port for network collection
streamsize=size # Max collection size per hour (eg. 1024 or 1K : K, M or G)
# Command to get cpu usage information from OS
cpu=vmstat 2 2 | tail -1 | awk '{printf("%d(usr+sys)",$14+$15);}'
# Command to get IO usage information from OS
io=vmstat 2 2 | tail -1 | awk '{printf("%d(bi+bo)",$10+$11);}'
# Ordering of information in sessions screen
# Column order for the session screen (option l)
sessions=0,1,18,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,19,20,21,22,23
# Column order for the tables screen (option T)
tables=0,1,2,4,3,5,6,7
# Column order for the tablespaces screen (option t).
# The display will be sorted in ascending order on column #22
tablespaces=0,1,18,2,3,4,5,6,7,8, sort=22a
# Column order for the bufferpool screen (option b)
bufferpools=0,1,18,2,3,4,5,6,7,8,9,10
# Column order for the Dynamic SQL screen (option D)
dynsql=0,1,18,2,3,4,5,6,7,8,9
statements=0,1
locks=0,1
utilities=0 # contains the default column and sort order for the utility screen
federation=0,2,4  # contains the default column and sort order for the federation screen

# User defined commands
shell P=top
function N=date && netstat -t tcp
```

Chapter 9. Switch-based monitoring concepts

System monitor switches

System monitor switches control how snapshot monitors and some event monitors collect data.

Note: These system monitor switches do not affect the unit of work event monitor and locking event monitor, which were introduced in DB2 Version 9.7.

The snapshot monitor and some event monitors report data collected by the system monitor. Collecting system monitor data introduces processing overhead for the database manager. For example, in order to calculate the execution time of SQL statements, the database manager must make calls to the operating system to obtain timestamps before and after the execution of every statement. These types of system calls are generally expensive. Another form of overhead incurred by the system monitor is increased memory consumption. For every monitor element tracked by the system monitor, the database manager uses its memory to store the collected data.

In order to minimize the overhead involved in maintaining monitoring information, monitor switches control the collection of potentially expensive data by the database manager. Each switch has only two settings: ON or OFF. If a monitor switch is OFF, the monitor elements under that switch's control do not collect any information. There is a considerable amount of basic monitoring data that is not under switch control, and will always be collected regardless of switch settings.

Each monitoring application has its own logical view of the monitor switches (and the system monitor data). Upon startup each application inherits its monitor switch settings from the dft_monswitches parameters in the database manager configuration file (at the instance level). A monitoring application can alter its monitor switch settings with the UPDATE MONITOR SWITCHES USING MONSWITCH OFF/ON command. The MONSWITCH parameter holds values found in the Monitor Switch column in the Snapshot Monitor Switches table below. Changes to the switch settings at the application level only affect the application from where the switch was changed.

Instance-level monitor switches can be changed without stopping the database management system. To do this use the UPDATE DBM CFG USING DBMSWITCH OFF/ON command. The `DBMSWITCH` parameter holds values from the DBM Parameter column in the Snapshot Monitor Switches table below. This dynamic updating of switches requires that the application performing the update be explicitly attached to the instance for the updates to dynamically take effect. Other existing snapshot applications will not be affected by a dynamic update. New monitoring applications will inherit the updated instance-level monitor switch settings. For an existing monitoring application to inherit the new default monitor switch values, it must terminate and re-establish its attachment. Updating the switches in the database manager configuration file will update the switches for all partitions in a partitioned database.

The database manager keeps track of all the snapshot monitoring applications and their switch settings. If a switch is set to ON in one application's configuration,

© Copyright IBM Corp. 1993, 2012

then the database manager always collects that monitor data. If the same switch is then set to OFF in the application's configuration, then the database manager will still collect data as long as there is at least one application with this switch turned ON.

The collection of time and timestamp elements is controlled by the TIMESTAMP switch. Turning this switch OFF (it is ON by default) instructs the database manager to skip any timestamp operating system calls when determining time or timestamp-related monitor elements. Turning this switch OFF becomes important as CPU utilization approaches 100%. When this occurs, the performance degradation caused by issuing timestamps increases dramatically. For monitor elements that can be controlled by the TIMESTAMP switch and another switch, if either of the switches is turned OFF, data is not collected. Therefore, if the TIMESTAMP switch is turned OFF, the overall cost of data under the control of other monitor switches is greatly reduced.

Event monitors are not affected by monitor switches in the same way as snapshot monitoring applications. When an event monitor is defined, it automatically turns ON the instance level monitor switches required by the specified event types. For example, a deadlock event monitor will automatically turn ON the LOCK monitor switch. The required monitor switches are turned ON when the event monitor is activated. When the event monitor is deactivated, the monitor switches are turned OFF.

The TIMESTAMP monitor switch is not set automatically by event monitors. It is the only monitor switch that controls the collection of any monitor elements belonging to event monitor logical data groupings. If the TIMESTAMP switch is OFF, most of the timestamp and time monitor elements collected by event monitors will not be collected. These elements are still written to the specified table, file, or pipe, but with a value of zero.

Table 32. Snapshot Monitor Switches

Monitor Switch	DBM Parameter	Information Provided
BUFFERPOOL	DFT_MON_BUFPOOL	Number of reads and writes, time taken
LOCK	DFT_MON_LOCK	Lock wait times, deadlocks
SORT	DFT_MON_SORT	Number of heaps used, sort performance
STATEMENT	DFT_MON_STMT	Start/stop time, statement identification
TABLE	DFT_MON_TABLE	Measure of activity (rows read/written)
UOW	DFT_MON_UOW	Start/end times, completion status
TIMESTAMP	DFT_MON_TIMESTAMP	Timestamps

Before capturing a snapshot or using an event monitor, you must determine what data you need the database manager to gather. If you want any of the following special types of data to be collected in a snapshot, you will need to set the appropriate monitor switches.

- Buffer pool activity information
- Lock, lock wait, and time related lock information

- Sorting information
- SQL statement information
- Table activity information
- Times and timestamp information
- Unit of work information

The switches corresponding to the above information types are all OFF by default, except for the switch corresponding to times and timestamp information, which is ON by default.

Event monitors are only affected by the time and timestamp information switch. All other switch settings have no effect on the data collected by event monitors.

Setting system monitor switches from the CLP

System monitor switches control the collection of data by the system monitor. By setting certain monitor switches to ON, you can collect specific types of monitor data.

Before you begin

The application performing any monitor switch updates must have an instance attachment. You must have one of SYSADM, SYSCTRL, SYSMAINT, or SYSMON authority to use the following commands:

- UPDATE MONITOR SWITCHES
- GET MONITOR SWITCHES
- GET DATABASE MANAGER MONITOR SWITCHES

You must have SYSADM authority to use the UPDATE DBM CFG command.

About this task

Procedure

- To activate any of the local monitor switches, use the UPDATE MONITOR SWITCHES command. The switches will remain active until the application (CLP) detaches, or until they are deactivated with another UPDATE MONITOR SWITCHES command. The following example updates all of the local monitor switches to be ON:

  ```
  db2 update monitor switches using BUFFERPOOL on LOCK on
      SORT on STATEMENT on TIMESTAMP on TABLE on UOW on
  ```

- To deactivate any of the local monitor switches, use the UPDATE MONITOR SWITCHES command. The following example updates all of the local monitor switches to be OFF:

  ```
  db2 update monitor switches using BUFFERPOOL off, LOCK off,
      SORT off, STATEMENT off, TIMESTAMP off, TABLE off, UOW off
  ```

  The following is an example of the output you would expect to see after issuing the above UPDATE MONITOR SWITCH command:

  ```
           Monitor Recording Switches

  Switch list for db partition number 1
  Buffer Pool Activity Information (BUFFERPOOL) = OFF
  Lock Information                         (LOCK) = OFF
  Sorting Information                      (SORT) = OFF
  SQL Statement Information           (STATEMENT) = OFF
  ```

```
Table Activity Information            (TABLE) = OFF
Unit of Work Information                (UOW) = OFF
Get timestamp information         (TIMESTAMP) = OFF
```

- It is also possible to manipulate the monitor switches at the database manager level. This involves changing the dft_monswitches parameters in the database manager configuration file, using the UPDATE DBM CFG command. In the following example, only lock switch controlled information is to be collected in addition to the basic information.

  ```
  db2 update dbm cfg using DFT_MON_LOCK on
  ```

  Whenever a monitoring application is started, it inherits its monitor switch settings from the database manager. Any changes to the database manager's monitor switch settings will not impact any running monitoring applications. Monitoring applications must reattach themselves to the instance to pick up any changes to monitor switch settings.
- For partitioned database systems, you can set monitor switches specifically for a certain partition, or globally for all partitions.
  1. To set a monitor switch (for example, BUFFERPOOL) for a specific partition (for example, partition number 3), issue the following command:

     ```
     db2 update monitor switches using BUFFERPOOL on
         at dbpartitionnum 3
     ```
  2. To set a monitor switch (for example, SORT) for all partitions, issue the following command:

     ```
     db2 update monitor switches using SORT on global
     ```
- To check the status of the local monitor switches use the GET MONITOR SWITCHES command.

  ```
  db2 get monitor switches
  ```
- For partitioned database systems, you can view the monitor switch settings specifically for a certain partition, or globally for all partitions.
  1. To view the monitor switch settings for a specific partition (for example, partition number 2), issue the following command:

     ```
     db2 get monitor switches at dbpartitionnum 2
     ```
  2. To view the monitor switch settings for all partitions, issue the following command:

     ```
     db2 get monitor switches global
     ```
- To check the status of the monitor switches at the database manager level (or instance level) use the GET DATABASE MANAGER MONITOR SWITCHES command. This command will show the overall switch settings for the instance being monitored.

  ```
  db2 get database manager monitor switches
  ```

  The following is an example of the output you should expect to see after issuing the above command:

  ```
           DBM System Monitor Information Collected

  Switch list for db partition number 1
  Buffer Pool Activity Information (BUFFERPOOL) = OFF
  Lock Information                        (LOCK) = ON    10-25-2001 16:04:39
  Sorting Information                     (SORT) = OFF
  SQL Statement Information          (STATEMENT) = OFF
  Table Activity Information             (TABLE) = OFF
  Unit of Work Information                 (UOW) = OFF
  Get timestamp information          (TIMESTAMP) = OFF
  ```

Results

Now that you have set the desired monitor switches and confirmed the switch settings, you are ready to capture and collect monitor data.

Setting system monitor switches from a client application

System monitor switches control the collection of data by the system monitor. By setting certain monitor switches to ON, you can collect specific types of monitor data.

Before you begin

The application performing any monitor switch updates must have an instance attachment. You must have SYSADM, SYSCTRL, SYSMAINT, or SYSMON authority to use the db2MonitorSwitches API.

Procedure

1. Include the following DB2 libraries: sqlutil.h and db2ApiDf.h. These are found in the `include` subdirectory under `sqllib`.

   ```
   #include <sqlutil.h>
   #include <db2ApiDf.h>
   #include <string.h>
   #include <sqlmon.h>
   ```

2. Set switch lists buffer unit size to 1 KB.

   ```
   #define SWITCHES_BUFFER_UNIT_SZ 1024
   ```

3. Initialize the sqlca, db2MonitorSwitches, and sqlm_recording_group structures. Also, initialize a pointer to contain the switch lists buffer, and establish the buffer's size.

   ```
   struct sqlca sqlca;
   memset (&sqlca, '\0', sizeof(struct sqlca));
   db2MonitorSwitchesData switchesData;
   memset (&switchesData, '\0', sizeof(switchesData));
   struct sqlm_recording_group switchesList[SQLM_NUM_GROUPS];
   memset(switchesList, '\0', sizeof(switchesList));
   sqluint32 outputFormat;
   static sqluint32 switchesBufferSize = SWITCHES_BUFFER_UNIT_SZ;
   char *switchesBuffer;
   ```

4. Initialize the buffer, which is to hold the switch list output.

   ```
   switchesBuffer = (char *)malloc(switchesBufferSize);
   memset(switchesBuffer, '\0', switchesBufferSize));
   ```

5. To alter the state of the local monitor switches, alter the elements in the sqlm_recording_group structure (named switchesList as indicated in the previous step). For a monitor switch to be turned on, the parameter input_state is to be set to SQLM_ON. For a monitor switch to be turned off, the parameter input_state must be set to SQLM_OFF.

   ```
   switchesList[SQLM_UOW_SW].input_state = SQLM_ON;
   switchesList[SQLM_STATEMENT_SW].input_state = SQLM_ON;
   switchesList[SQLM_TABLE_SW].input_state = SQLM_ON;
   switchesList[SQLM_BUFFER_POOL_SW].input_state = SQLM_OFF;
   switchesList[SQLM_LOCK_SW].input_state = SQLM_OFF;
   switchesList[SQLM_SORT_SW].input_state = SQLM_OFF;
   switchesList[SQLM_TIMESTAMP_SW].input_state = SQLM_OFF;
   switchesData.piGroupStates = switchesList;
   switchesData.poBuffer = switchesBuffer;
   switchesData.iVersion = SQLM_DBMON_VERSION9_5;
   switchesData.iBufferSize = switchesBufferSize;
   ```

```
switchesData.iReturnData = 0;
switchesData.iNodeNumber = SQLM_CURRENT_NODE;
switchesData.poOutputFormat = &outputFormat;
```

Note: SQLM_TIMESTAMP_SW is unavailable if iVersion is less than SQLM_DBMON_VERSION8.

6. To submit the changes to switch settings, call the db2MonitorSwitches() function. Pass the db2MonitorSwitchesData structure (named switchesData in this example) as a parameter to the db2MonitorSwitches API. The switchesData contains the sqlm_recording_group structure as a parameter.

   ```
   db2MonitorSwitches(db2Version810, &switchesData, &sqlca);
   ```

7. Process the switch list data stream from the switch list buffer.
8. Clear the switch list buffer.

   ```
   free(switchesBuffer);
   free(pRequestedDataGroups);
   ```

Results

Now that you have set the desired monitor switches and confirmed the switch settings, you are ready to capture and collect monitor data.

System monitor switches self-describing data stream

After you update or view the current system monitor switch settings with the db2MonitorSwitches API, the API returns the switch settings as a self-describing data stream. Figure 14 on page 289 shows the structure of the switch list information that may be returned for a partitioned database environment.

Note:

1. Descriptive names are used for the identifiers in the examples and tables. These names are prefixed by **SQLM_ELM_** in the actual data stream. For example, db_event would appear as SQLM_ELM_DB_EVENT in the event monitor output. Types are prefixed with **SQLM_TYPE_** in the actual data stream. For example, headers appear as SQLM_TYPE_HEADER in the data stream.
2. For global switch requests the partition order of the returned information can be different in each switch request. In this case, a partition id is included in the data stream.

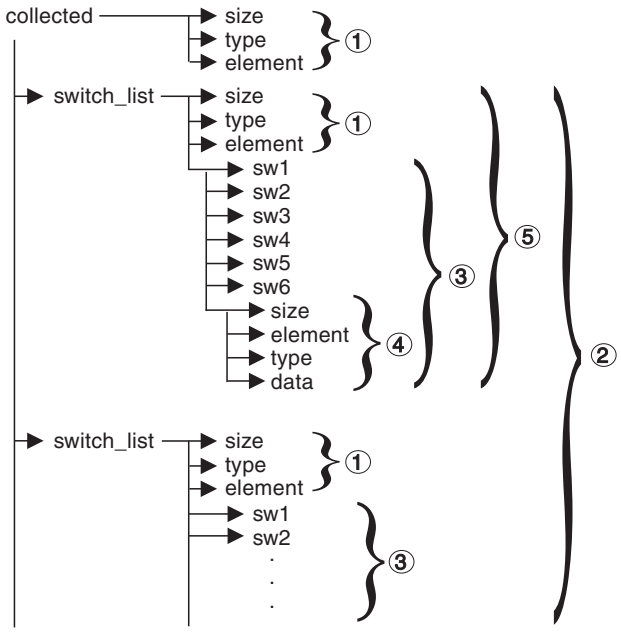


Figure 14. Switch List Monitor Data Stream

1. Each logical data group begins with a header that indicates its size and name. This size does not include the volume of data taken up by the header itself.
2. Size in the collected header returns the total size of all monitor switch lists for all partitions.
3. The size element in switch list header indicates the size of switch data for that partition.
4. Switch information is self-describing.
5. For a non-partitioned database, the switch settings for the stand alone partition are returned. That is, only one switch list is returned.

Database system monitor data organization

The system monitor collects and stores information that you can access using interfaces to the snapshot monitor and some event monitors. The database system monitor stores information it collects in entities called *monitor elements* (these were previously known as data elements). Each monitor element stores information regarding one specific aspect of the state of the database system. In addition, monitor elements are identified by unique names and store a certain type of information.

The following are the available element types used by the system monitor in which monitor elements store data:

Counter
Counts the number of times an activity occurs. Counter values increase during monitoring. Most counter elements can be reset.

Gauge Indicates the current value for an item. Gauge values can go up and down depending on database activity (for example, the number of locks held). Gauge elements can not be reset.

Watermark
Indicates the highest (maximum) or lowest (minimum) value an element has reached since monitoring was started. Watermark elements can not be reset.

Information
Provides reference-type details of your monitoring activities. This can include items such as partition names, aliases, and path details. Information elements can not be reset.

Timestamp
Indicates the date and time that an activity took place by providing the number of seconds and microseconds that have elapsed since January 1, 1970. For the snapshot monitor and event monitors, the collection of timestamp elements is controlled by the TIMESTAMP monitor switch. While this switch is on by default, you should turn it off for performance reasons if CPU utilization on the database instance approaches 100%. Timestamp elements can not be reset.

A value of 0 for the timestamp element means "not available". If you attempt to import this data, such a value will generate an out of range error (SQL0181). To avoid this error, update the value to any valid timestamp value before exporting the data.

Time Returns the number of seconds and microseconds spent on an activity. For the snapshot monitor and event monitors, the collection of most time elements is controlled by the TIMESTAMP monitor switch. While this switch is on by default, you should turn it off for performance reasons if CPU utilization on the database instance approaches 100%. Some time elements can be reset.

Monitor elements collect data for one or more logical data groups. A logical data group is a collection of monitor elements that gather database system monitoring information for a specific scope of database activity. Monitor elements are sorted in logical data groups based on the levels of information they provide. For example, while snapshot monitoring, the Total Sort Time monitor element returns database (dbase), application (appl), and statement (stmt) information; hence, it appears in each of the logical data groups listed in parentheses.

Although many monitor elements are used by both the snapshot monitor and event monitors, they each use a distinct set of logical data groups. This is because the scopes of database activity for which you can capture a snapshot differ from those for which you can collect event data. Practically speaking, the overall set of monitor elements accessible from the snapshot monitor is different from those accessible from event monitors.

Counter status and visibility

Among the monitor elements collected by the system monitor are several accumulating counters. These counters are incremented during the operation of the database or database manager, for example, every time an application commits a transaction.

Counters are initialized when their applicable object becomes active. For instance, the number of buffer pool pages read for a database (a basic monitor element) is set to zero when the database is activated.

Some counters that can be collected by the system monitor are controlled by monitor switches. If a particular monitor switch is off, the monitor elements under its control do not collect data. When a monitor switch is turned on, all the associated counters are reset to zero.

Counters returned by event monitors are reset to zero when the event monitor is activated.

Event monitor counting represents a count since one of the following starting points:

- Event monitor startup, for database, table space, and tables.
- Event monitor startup, for existing connections.
- Application connection, for connections made after the monitor was started.
- Start of the next transaction (unit of work) or statement after the monitor was started.
- Occurrence of a deadlock after the monitor was started.

Each event monitor and any monitoring application (an application using the snapshot monitor APIs) has its own logical view of the system monitor data. This means that when counters are reset or initialized, it only affects the event monitor or application that reset or initialized them. Event monitor counters cannot be reset, except by turning the event monitor off, and then on again. An application taking snapshots can reset its view of the counters at any time by using the RESET MONITOR command.

If you start a statement event monitor after a statement starts, the monitor will start collecting information when the next SQL statement starts. As a result, the event monitor will not return information about statements that the database manager is executing when the monitor was started. This is also true for transaction information.

System monitor output: the self-describing data stream

Aside from presenting system monitor data on screen or storing it in SQL tables, you can develop a client application to process it. The system monitor returns monitor data via a self-describing data stream for both the snapshot monitor and event monitor. In a snapshot monitoring application you can call the snapshot APIs to capture a snapshot and then directly process the data stream.

Processing event monitor data is different, in that the event data is sent to the application at the pace database events occur. For a pipe event monitor, the application waits for event data to arrive, and then processes it when it does. For a file event monitor, the application parses event files, thus processing event records in batches.

This self-describing data stream allows you to parse through the returned data one element at a time. This opens up numerous monitoring possibilities, including looking for information regarding a particular application or a specific database state.

The returned monitor data is in the following format:

size The size (in bytes) of the data stored in the monitor element or logical data grouping. In the case of a logical data grouping, this is the size of all data in the logical group. For example, the database logical grouping (*db*)

contains individual monitor elements (such as *total_log_used*) along with other logical data groupings, such as rollforward information (*rollforward*). This does not include the size taken up by the 'size', 'type', and 'element' information.

type The type of element stored in the data (for example, variable length string or signed 32 bit numeric value). An element type of *header* refers to a logical data grouping for an element.

element id
The identifier for the monitor element that was captured by the monitor. In the case of a logical data grouping, this is the identifier for the group (for example, *collected*, *dbase*, or *event_db*).

data The value collected by a monitor for a monitor element. In the case of a logical data grouping, the data is composed of the monitor elements belonging to it.

All timestamps in monitor elements are returned in two unsigned 4 byte monitor elements (seconds and microseconds). These represent the number of seconds since January 1, 1970 in GMT time.

The size element of strings in monitor elements represents the actual size of data for the string element. This size does not include a null terminator, as the strings are not null terminated.

Memory requirements for monitor data

The memory required for monitor data is allocated from the monitor heap. Monitor heap size is controlled by the **mon_heap_sz** database configuration parameter. This parameter has a default value of AUTOMATIC, meaning that the monitor heap can increase as needed until the instance_memory limit is reached.

If you configure the **mon_heap_sz** parameter manually, consider the following factors:

- The number of monitoring applications
- The number and nature of event monitors
- The monitor switches set
- The level of database activity

Consider increasing the value for the **mon_heap_sz** parameter if monitor commands fail with an SQLCODE of -973.

The following formula provides an approximation of the number of pages required for the monitor heap:

```
(Memory used by applications            +
 Memory used by event monitors          +
 Memory used by monitoring applications +
 Memory used by Gateway applications)      / 4096
```

Memory used by each application

- If the STATEMENT switch is off, zero
- If the STATEMENT switch is on:
  - Add 400 bytes for each statement being run at the same time. (That is, the number of open cursors that an application might have). This is *not* the cumulative total of statements an application has run.

  - If a partitioned database, add the following for each statement:
    - 200 bytes * (average # of subsections)
- If the application has issued sqleseti() info, add the sizes of the userid, applname, workstation name and accounting string.

Memory used by each event monitor

For each event monitor of type ACTIVITIES:

- 3500 bytes
- If the event monitor is for type TABLES, add 36K * (number of CPU cores + 1)
- If the event monitor is for type FILE or PIPE, add 2K * (number of CPU cores + 1)

If you expect a heavy volume, add 250 megabytes for event records. Otherwise add a fraction that depends on the expected amount of work.

For each event monitor of type LOCKING or UOW:

- 3500 bytes
- 3K * (number of CPU cores + 1)

If you expect a heavy volume, add 250 megabytes for event records. Otherwise add a fraction that depends on the expected amount of work.

For each event monitor of the following type: DATABASE, TABLES, TABLESPACES, BUFFERPOOLS, CONNECTIONS, DEADLOCK:

- 4100 bytes
- 2 * BUFFERSIZE
- If the event monitor is written to a file, add 550 bytes.
- If the event monitor is for type DATABASE:
  - add 6000 bytes
  - add 100 bytes for each statement in the statement cache
- If the event monitor is for type TABLES:
  - add 1500 bytes
  - add 70 bytes for each table accessed
- If the event monitor is for type TABLESPACES:
  - add 450 bytes
  - add 350 bytes for each table space
- If the event monitor is for type BUFFERPOOLS:
  - add 450 bytes
  - add 340 bytes for each buffer pool
- If the event monitor is for type CONNECTIONS:
  - add 1500 bytes
  - for each connected application:
    - add 750 bytes
  - remember to add the value from “Memory used by each application” on page 292.
- If an event monitor is of type DEADLOCK:
  - and the WITH DETAILS HISTORY is running:

- add X*475 bytes times the maximum number of concurrent applications you expect to be running, where X is the expected maximum number of statements in your application's unit of work.

– and the WITH DETAILS HISTORY VALUES is running:

- also add X*Y bytes times the maximum number of concurrent applications you expect to be running, where Y is the expected maximum size of parameter values being bound into your SQL statements.

Memory used by each monitoring application

- 250 bytes
- For each database being reset:
  - 350 bytes
  - Add 200 bytes for each REMOTE database.
  - If the SORT switch is on, add 25 bytes.
  - If the LOCK switch is on, add 25 bytes.
  - If the TABLE switch is on:
    - add 600 bytes
    - add 75 bytes per table accessed
  - If the BUFFERPOOL switch is on:
    - add 300 bytes
    - add 250 bytes per table space accessed
    - add 250 bytes per buffer pool accessed
  - If the STATEMENT switch is on:
    - add 2100 bytes
    - add 100 bytes per statement
  - For each application connected to the database:
    - add 600 bytes
    - add 200 bytes for every REMOTE database the application is connected to
    - if the SORT switch is on, add 25 bytes
    - if the LOCK switch is on, add 25 bytes
    - if the BUFFERPOOL switch is on, add 250 bytes
- For each DCS database being reset:
  - add 200 bytes for the database
  - add 200 bytes for each application connected to the database
  - if the STATEMENT switch is ON, Transmission level data must be reset:
    - for each database, add 200 bytes for each transmission level
    - for each application, add 200 bytes for each transmission level

Memory used by gateway applications

- 250 bytes for each host database (even if all switches are off)
- 400 bytes for each application (even if all switches are off)
- If the STATEMENT switch is on:
  - For each application, add 200 bytes for each statement being run at the same time (That is, the number of open cursors that an application might have). This is NOT the cumulative total of statements an application has run.
  - Transmission level data must be accounted for:
    - for each database, add 200 bytes for each transmission level

- for each application, add 200 bytes for each transmission level

- If the UOW switch is on:
  - add 50 bytes for each application
- For each application using a TMDB (for SYNCPOINT TWOPHASE activity):
  - add 20 bytes plus the size of the XID itself
- For any application that has issued sqleseti to set client name, app name, wkstn or accounting:
  - add 800 bytes plus the size of the accounting string itself

Monitoring buffer pool activity

Monitoring buffer pool activity in your database can help you identify opportunities to improve database performance.

The database server reads and updates all data from a buffer pool. Data is copied from disk to a buffer pool as it is required by applications.

Pages are placed in a buffer pool:

- by the agent. This is synchronous I/O.
- by the I/O servers (prefetchers). This is asynchronous I/O.

Pages are written to disk from a buffer pool:

- by the agent, synchronously
- by page cleaners, asynchronously

If the server needs to read a page of data, and that page is already in the buffer pool, then the ability to access that page is much faster than if the page had to be read from disk. It is desirable to **hit** as many pages as possible in the buffer pool. Avoiding disk I/O is an important factor in database performance, therefore proper configuration of the buffer pools is one of the most important considerations for performance tuning.

The buffer pool hit ratio indicates the percentage of time that the database manager did not need to load a page from disk in order to service a page request because the page was already in the buffer pool. The greater the buffer pool hit ratio, the lower the frequency of disk I/O.

The overall buffer pool hit ratio can be calculated as follows:

```
    1 - ((pool_data_p_reads + pool_xda_p_reads +
pool_index_p_reads + pool_temp_data_p_reads
+ pool_temp_xda_p_reads + pool_temp_index_p_reads )
     / (pool_data_l_reads + pool_xda_l_reads + pool_index_l_reads +
     pool_temp_data_l_reads + pool_temp_xda_l_reads
     + pool_temp_index_l_reads )) * 100%
```

This calculation takes into account all of the pages (index and data) that are cached by the buffer pool.

You can also use the BP_HITRATIO administrative view as a convenient method of monitoring the hit ratio for your buffer pools.

For a large database, increasing the buffer pool size may have minimal effect on the buffer pool hit ratio. Its number of data pages may be so large, that the

statistical chances of a hit are not improved by increasing its size. Instead, you might find that tuning the index buffer pool hit ratio achieves the desired result. This can be achieved using two methods:

1. Split the data and indexes into two different buffer pools and tune them separately.
2. Use one buffer pool, but increase its size until the index hit ratio stops increasing. The index buffer pool hit ratio can be calculated as follows:

```
(1 - ((pool_index_p_reads) / (pool_index_l_reads))) * 100%
```

The first method is often more effective, but because it requires indexes and data to reside in different table spaces, it may not be an option for existing databases. It also requires tuning two buffer pools instead of one, which can be a more difficult task, particularly when memory is constrained.

You should also consider the impact that prefetchers may be having on the hit ratio. Prefetchers read data pages into the buffer pool anticipating their need by an application (asynchronously). In most situations, these pages are read just before they are needed (the desired case). However, prefetchers can cause unnecessary I/O by reading pages into the buffer pool that will not be used. For example, an application starts reading through a table. This is detected and prefetching starts, but the application fills an application buffer and stops reading. Meanwhile, prefetching has been done for a number of additional pages. I/O has occurred for pages that will not be used and the buffer pool is partially taken up with those pages.

Page cleaners monitor the buffer pool and asynchronously write pages to disk. Their goals are:

- Ensure that agents will always find free pages in the buffer pool. If an agent does not find free pages in the buffer pool, it must clean them itself, and the associated application will have a poorer response.
- Speed database recovery, if a system crash occurs. The more pages that have been written to disk, the smaller the number of log file records that must be processed to recover the database.

Although dirty pages are written out to disk, the pages are not removed from the buffer pool right away, unless the space is needed to read in new pages.

Note: Buffer pool information is typically gathered at a table space level, but the facilities of the database system monitor can roll this information up to the buffer pool and database levels. Depending on your type of analysis, you may need to examine this data at any or all of these levels.

Database system monitor interfaces

Monitoring task	API
Capturing a snapshot	db2GetSnapshot
Converting the self-describing data stream	db2ConvMonStream
Displaying the database system monitor switches	db2MonitorSwitches
Estimating the size of a snapshot	db2GetSnapshotSize
Get/update monitor switches	db2MonitorSwitches
Resetting monitor counters	db2ResetMonitor

Monitoring task	API
Updating the database system monitor switches	db2MonitorSwitches

Monitoring task	CLP Command
Analyzing event monitor output with a GUI tool	db2eva
Capturing a snapshot	GET SNAPSHOT
Displaying the database manager monitor switches	GET DATABASE MANAGER MONITOR SWITCHES
Displaying the monitoring application's monitor switches	GET MONITOR SWITCHES
Formatting the event monitor trace	db2evmon
Generating sample SQL for write-to-table CREATE EVENT MONITOR statements	db2evtbl
Listing the active databases	LIST ACTIVE DATABASES
Listing the applications connected to a database	LIST APPLICATIONS
Listing the DCS applications	LIST DCS APPLICATIONS
Resetting monitor counters	RESET MONITOR
Updating the database system monitor switches	UPDATE MONITOR SWITCHES

Monitoring task	SQL Statement
Activating an event monitor	SET EVENT MONITOR STATE
Creating an event monitor	CREATE EVENT MONITOR
Deactivating an event monitor	SET EVENT MONITOR STATE
Removing an event monitor	DROP
Writing event monitor values	FLUSH EVENT MONITOR

Monitoring task	SQL Function
Determining the state of an event monitor	EVENT_MON_STATE scalar function
Getting a database manager level snapshot	SNAPDBM administrative view and SNAP_GET_DBM_V95 table function
Getting the current monitor switch settings at the database manager level	SNAPSWITCHES administrative view and SNAP_GET_SWITCHES table function
Getting a fast communication manager snapshot	SNAPFCM administrative view and SNAP_GET_FCM table function
Getting a fast communication manager snapshot for a given partition	SNAPFCM_PART administrative view and SNAP_GET_FCM_PART table function
Getting a database level snapshot	SNAPDB administrative view and SNAP_GET_DB_V95 table function
Getting an application level snapshot	SNAPAPPL administrative view and SNAP_GET_APPL_V95 table function
Getting an application level snapshot	SNAPAPPL_INFO administrative view and SNAP_GET_APPL_INFO_V95 table function

Monitoring task	SQL Function
Getting an application level snapshot for lock wait information	SNAPLOCKWAIT administrative view and SNAP_GET_LOCKWAIT table function
Getting an application level snapshot for statement information	SNAPSTMT administrative view and SNAP_GET_STMT table function
Getting an application level snapshot for agent information	SNAPAGENT administrative view and SNAP_GET_AGENT table function
Getting an application level snapshot for subsection information	SNAPSUBSECTION administrative view and SNAP_GET_SUBSECTION table function
Getting a buffer pool level snapshot	SNAPBP administrative view and SNAP_GET_BP_V95 table function
Getting a table space level snapshot	SNAPTBSP administrative view and SNAP_GET_TBSP_V91 table function
Getting a table space level snapshot for configuration information	SNAPTBSP_PART administrative view and SNAP_GET_TBSP_PART_V91 table function
Getting a table space level snapshot for container information	SNAPCONTAINER administrative view and SNAP_GET_CONTAINER_V91 table function
Getting a table space level snapshot for quiescer information	SNAPTBSP_QUIESCER administrative view and SNAP_GET_TBSP_QUIESCER table function
Getting a table space level snapshot for the ranges of a table space map	SNAPTBSP_RANGE administrative view and SNAP_GET_TBSP_RANGE table function
Getting a table level snapshot	SNAPTAB administrative view and SNAP_GET_TAB_V91 table function
Getting a lock level snapshot	SNAPLOCK administrative view and SNAP_GET_LOCK table function
Getting a snapshot of SQL statement cache information	SNAPDYN_SQL administrative view and SNAP_GET_DYN_SQL_V95 table function

Chapter 10. Deprecated monitoring tools

Health monitor

Monitoring database health

Introduction to the health monitor

The health monitor is a server-side tool that adds a management-by-exception capability by constantly monitoring the health of an instance and active databases. The health monitor can also alert a database administrator (DBA) of potential system health issues.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

The health monitor proactively detects issues that might lead to hardware failure, or to unacceptable system performance or capability. The proactive nature of the health monitor enables users to address an issue before it becomes a problem that affects system performance.

The health monitor checks the state of your system using health indicators to determine whether an alert should be issued. Preconfigured actions can be taken in response to alerts. The health monitor can also log alerts in the administration notification log and send notifications by email or pager. This management-by-exception model frees up valuable DBA resources by generating alerts to potential system health issues without requiring active monitoring.

The health monitor periodically gathers information about the health of the system with a minimal impact to overall performance. It does not turn on any snapshot monitor switches to collect information.

Health indicators:

The health monitor uses health indicators to evaluate the health of specific aspects of database manager performance or database performance. A health indicator measures the health of some aspect of a particular class of database objects, such as table spaces. Criteria are applied to the measurement to determine healthiness. The criteria applied depends on the type of health indicator. A determination of unhealthiness is based on the criteria generates an alert.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Three types of health indicators are returned by the health monitor:

- **Threshold-based** indicators are measurements that represent a statistic (on a continuous range of values) of the behavior of the object. Warning and alarm

© Copyright IBM Corp. 1993, 2012

threshold values define boundaries or zones for normal, warning, and alarm ranges. Threshold-based health indicators have three valid states: Normal, Warning, or Alarm.

- **State-based** indicators are measurements that represent a finite set of two or more distinct states of an object that defines whether the database object or resource is operating normally. One of the states is normal and all others are considered non-normal. State-based health indicators have two valid states: Normal, Attention.
- **Collection state-based** indicators are database-level measurements that represent an aggregate state or one or more objects within the database. Data is captured for each object in the collection and the highest severity of conditions among those objects is represented in the aggregated state. If one or more objects in the collection are in a state requiring an alert, the health indicator shows Attention state. Collection state-based health indicators have two valid states: Normal, Attention.

Health indicators exist at the instance, database, table space, and table space container level.

You can access health monitor information through the Health Center, the CLP, or APIs. You can configure health indicators through these same tools.

An alert is generated in response to either a change from a normal to a non-normal state or a change in the health indicator value to a warning or alarm zone that is based on defined threshold boundaries. There are three types of alerts: attention, warning, and alarm.

- For health indicators measuring distinct states, an attention alert is issued if a non-normal state is registered.
- For health indicators measuring a continuous range of values, threshold values define boundaries or zones for normal, warning and alarm states. For example, if the value enters the threshold range of values that defines an alarm zone, an alarm alert is issued to indicate that the problem needs immediate attention.

The health monitor will only send notification and run an action on the first occurrence of a particular alert condition for a given health indicator. If the health indicator stays in a particular alert condition, no further notification will be sent and no further actions will be run. If the health indicator changes alert conditions, or goes back to normal state and re-enters the alert condition, notification will be sent anew and actions will be run.

The following table shows an example of a health indicator at different refresh intervals and the health monitor response to the health indicator state. This example uses the default warning of 80% and alarm thresholds of 90%.

Table 33. Health indicator conditions at different refresh intervals

Refresh interval	Value of ts.ts_util (Table space utilization) health indicator	State of ts.ts_util health indicator	Health monitor response
1	80	warning	notification of warning is sent, actions for a warning alert condition are run
2	81	warning	no notification is sent, no actions are run

Table 33. Health indicator conditions at different refresh intervals (continued)

Refresh interval	Value of ts.ts_util (Table space utilization) health indicator	State of ts.ts_util health indicator	Health monitor response
3	75	normal	no notification is sent, no actions are run
4	85	warning	notification of warning is sent, actions for a warning alert condition are run
5	90	alarm	notification of alarm is sent, actions for an alarm condition are run

Health indicator process cycle:

The following diagram illustrates the evaluation process for health indicators. The set of steps runs every time the refresh interval for the specific health indicator elapses.

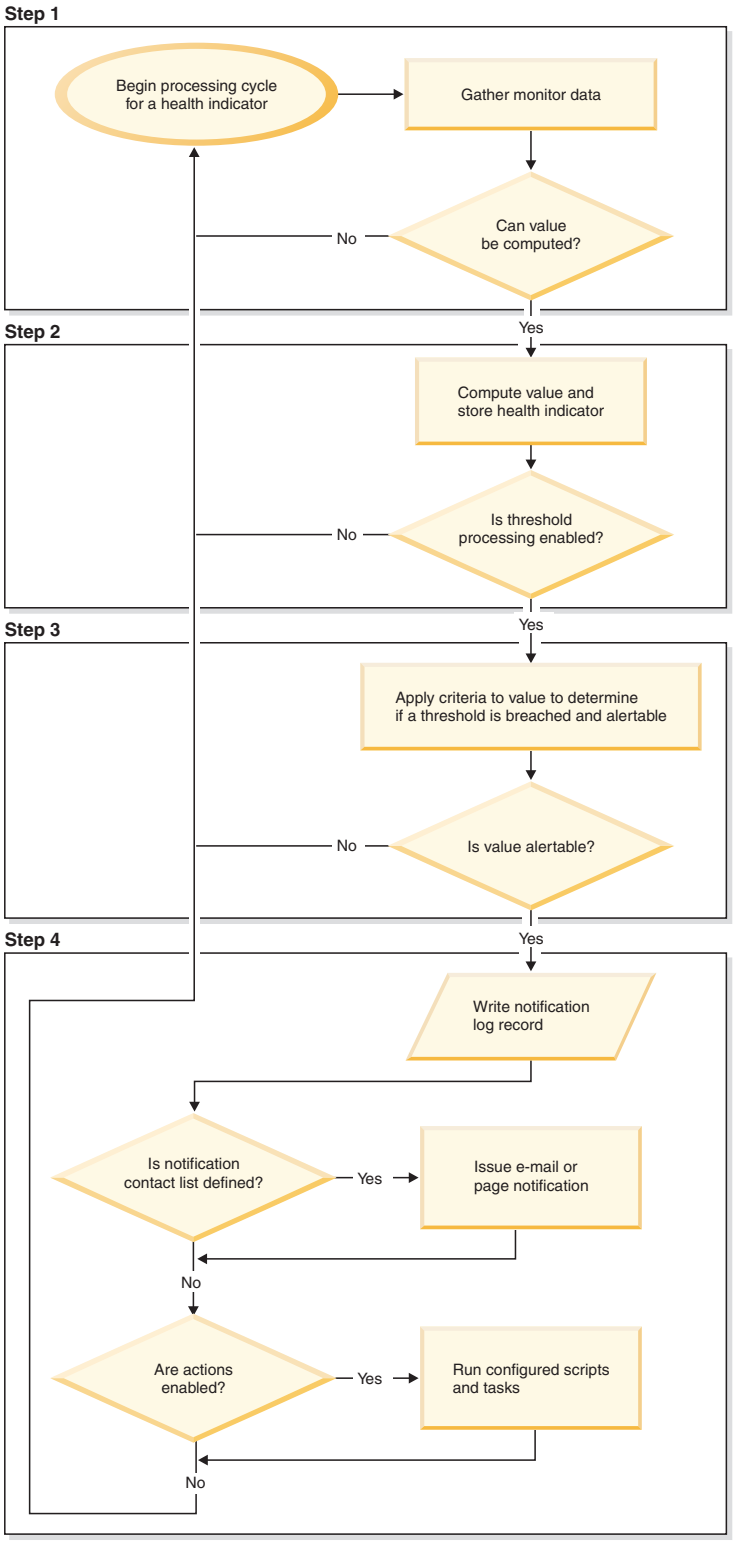


Figure 15. Health indicator process cycle

Note:

1. The NOTIFYLEVEL database manager configuration parameter controls whether alert notifications are sent to the DB2 administration notification log

and to any defined contacts. A minimum severity level of 2 is required for alarm notifications. A minimum severity level of 3 is required for warnings and attention alerts to be sent.

Enabling health alert notification:

To enable e-mail or pager notification when an alert is generated, you must set configuration parameters and specify contact information.

Before you begin

The DB2 Administration Server (DAS) must be running on the system where the contact list is located. For example, if the CONTACT_HOST configuration parameter is set to a remote system, the DAS must be running on the remote system in order for contacts to be notified of alerts.

About this task

To enable health alert notification:

Procedure

1. Specify the SMTP_SERVER parameter. The DAS configuration parameter, SMTP_SERVER, specifies the location of the mail server to use when sending both e-mail and pager notification messages. Omit this step if the system where the DB2 database is installed is enabled as an unauthenticated SMTP server.
2. Specify the CONTACT_HOST parameter. The DAS configuration parameter, CONTACT_HOST, specifies the remote location of the contact list for all instances on the local system. By setting this parameter, a single contact list can be shared between multiple systems. Omit this step if you want to keep the contact list on the local system where the DB2 database is installed.
3. Specify the default contact for health monitor notification. To enable e-mail or pager notification from the health monitor when an alert is generated, a default administration contact must be specified. If you choose not to provide this information, notification messages cannot be sent for alert conditions. You can provide the default administration contact information during installation, or you can defer the task until after installation is complete. If you choose to defer the task or want to add more contacts or groups to the notification list, you can specify contacts through the CLP, C APIs, or the Health Center:
   - **To specify contacts using the CLP:**
     To define an e-mail contact as the default for health monitor notification, issue the following commands:

     ```
     DB2 ADD CONTACT contact_name TYPE EMAIL ADDRESS
        email_address DESCRIPTION 'Default Contact'

     DB2 UPDATE NOTIFICATION LIST ADD CONTACT contact_name
     ```

     For complete syntax details, see the Command Reference.
   - **To specify contacts using C APIs:**
     The following C code excerpt illustrates how to define health notification contacts:

```
...
#include <db2ApiDf.h>

SQL_API_RC rc = 0;
struct db2AddContactData addContactData;
struct sqlca sqlca;

char* userid = "myuser";
char* password = "pwd";
char* contact = "DBA1";
char* email = "dba1@mail.com";
char* desc = "Default contact";

memset(&addContactData, '\0', sizeof(addContactData));
memset (&sqlca, '\0', sizeof(struct sqlca));

addContactData.piUserid = userid;
addContactData.piPassword = password;
addContactData.piName = contact;
addContactData.iType = DB2CONTACT_EMAIL;
addContactData.piAddress = email;
addContactData.iMaxPageLength = 0;
addContactData.piDescription = desc;

rc = db2AddContact(db2Version810, &addContactData, &sqlca);

if (rc == 0) {
   db2HealthNotificationListUpdate update;
   db2UpdateHealthNotificationListData data;
   db2ContactTypeData contact;

   contact.pName = contact;
   contact.contactType = DB2CONTACT_EMAIL;

   update.iUpdateType = DB2HEALTHNOTIFICATIONLIST_ADD;
   update.piContact = &contact;

   data.iNumUpdates = 1;
   data.piUpdates = &update;

   rc = db2UpdateHealthNotificationList (db2Version810, &data, &ca);
}
...
```

- **To specify contacts using the Health Center:**
  a. Right-click the instance for which you want to define the health notification list.
  b. Click **Configure**, then click **Alert Notification**. The Configure Health Alert Notification window opens.
  c. If contacts do not appear in the left side of the window in the **Available** list, click **Manage Contacts**. The Contacts window opens with the system name preselected.
  d. Click **Add Contact**. The Add Contact window opens.
  e. Define a contact by supplying a name and an e-mail address. Select **Address is for a pager** if the specified e-mail address is for a pager.
  f. Click **OK**.
  g. Close the Contacts window and return to the Configure Health Alert Notification window. The new contact now appears in the **Contacts available** list.

h. Move the contact to the **Health notification contact list** by clicking the right arrow button.
i. Click **OK** to include the contact in the health notification list.

Recommendation

If you are experiencing difficulties with notification, select **Troubleshoot** below the Health notification contact list. The Troubleshoot Health Alert Notification wizard opens.

Health Center overview

Use the Health Center to analyze and improve the health of DB2.

The following are examples of conditions that define what makes DB2 healthy:

- There are sufficient resources, such as free memory, table space containers, or logging storage, to accomplish tasks.
- Resources are used efficiently.
- Tasks complete within acceptable periods of time or without significant degradations in performance.
- Resources or database objects are not left indefinitely in unusable states.

Important: The Health Center has been deprecated in Version 9.7 and might be removed in a future release. Start using IBM Data Studio and IBM Optim™ tools. For a mapping between these recommended tools and Control Center tools, see "Table of recommended tools versus Control Center tools" in the *What's New for DB2 Version 9.7* book.

From the Health Center you can also open other centers and tools to help you investigate and maintain the health of your database.

To open the Health Center on Intel platforms, from the **Start** menu, click **Start** → **Programs** → **IBM DB2** → **Monitoring Tools** → **Health Center**.

To open the Health Center using the command line on Intel platforms, run the following command:

```
db2hc
```

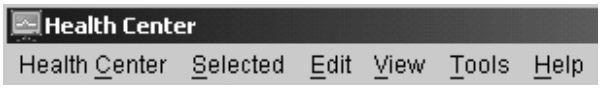
The following list categorizes some of the key tasks that you can do with the Health Center:

- "Enabling health alert notification" on page 303
  - Specifying contact settings and notification configuration parameters
  - Troubleshooting health alert notification
- "Configuring health indicators using the Health Center" on page 330
  - Enabling and disabling health indicator evaluation
  - Changing alert threshold and sensitivity settings
  - Running tasks and scripts when an alert occurs
- "Resolving health monitor alerts using the Health Center" on page 323
  - Using the Recommendation advisor to select and implement recommendations

The Health Center interface

The Health Center interface has the following elements that help you determine and resolve problems related to the overall health of your system.

Health Center Menu bar



Use the menu bar to work with objects in the Health Center, open other administration centers and tools, and access online help.

The Health Center menu bar contains the following menus:

Health Center toolbar

Use the toolbar icons below the menu bar to access other centers and tools, and to refresh the content view of the Health Center.

Toggle buttons

Use the toggle buttons to select the alert states that appear in the Navigation view. Each button corresponds to a minimum alert severity that a database object needs in order to appear in the view. Selecting a different button only affects the display and does not affect the object itself.

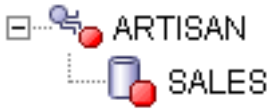
Shows objects in alarm state

Shows objects in alarm and warning states

Shows objects in any alert state: alarm, warning, attention, normal, and not monitored.

Shows all objects

Navigation view



Use the navigation view to display and work with instance and database objects. When you select an object in the Navigation view, the current alerts for that object and all its children are displayed in the Alerts view. To change the level of alert that object must have before the navigation view displays it, right click in the navigation view away from the listed objects. This will open a pop-up menu of alert levels. Select the alert levels that you want displayed. You can also choose what alert levels to display by clicking the toggle buttons.

Alerts view

Use the Alerts view to display and work with current alerts. The Alerts view displays those alerts that currently exist for the object and its children database objects selected in the navigation view. For example, if you select an instance, alerts display for the instance and all its databases and table spaces. If you select a database, alerts display for the database and all table

spaces for the database. Select and right-click one or more alerts in the Alerts view to invoke actions for them.

Alerts view toolbar

Use the toolbar below the Alerts view to tailor the view of alerts in the Alerts view to suit your needs.

Investigating alert conditions:

Health monitor

The health monitor captures information about the database manager, database, table space, and table space containers. The health monitor calculates health indicators based on data retrieved from database system monitor elements, the operating system, and DB2 database.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

The health monitor can only evaluate health indicators on a database and its objects when the database is active. You can keep the database active either by starting it with the **ACTIVATE DATABASE** command or by maintaining a permanent connection to the database.

The health monitor retains a maximum of 10 history records for each health indicator. This history is stored in the *instance_path*\hmonCache directory and is removed when the health monitor is stopped. The health monitor automatically prunes obsolete history records when the maximum number of records is reached.

Health monitor data is accessible through health snapshots. Each health snapshot reports the status for each health indicator based on its most recent refresh interval. The snapshots are useful for detecting existing database health problems and predicting potential poor health of the database environment. You can capture a health snapshot from the CLP, by using APIs in a C or C++ application, or by using the graphical administration tools.

Health monitoring requires an instance attachment. If an attachment to an instance was not established using the **ATTACH TO** command, then a default instance attachment to the local instance is created.

In partitioned database environments, snapshots can be taken at any partition of the instance, or globally using a single instance connection. A global snapshot aggregates the data collected at each partition and returns a single set of values.

Usage notes

The health monitor is supported on all editions of the DB2 database.

To start or stop the health monitor from the Health Center, right-click an instance in the Health Center navigational view and select **Start Health Monitor** or **Stop Health Monitor**.

On Windows, the service for the DB2 instance needs to run under an account with SYSADM authority. You can use the **-u** option on the **db2icrt** command, or use the Services folder on Windows and edit the Log On properties to use an account with administrator privilege.

The health monitor process runs as a DB2 fenced mode process. These processes appear as DB2FMP on Windows. On other platforms, the health monitor process appears as DB2ACD.

The DB2 Administration server must be running on the system where the health monitor resides for notifications to be sent and alert actions to be run. If remote scripts, tasks, or contact lists are used, the DB2 Administration server on the remote system must also be started.

The tools catalog database is required only for creating tasks. If you do not use alert task actions for any health indicator, the tools catalog database is not required by the health monitor.

If you fall back to DB2 UDB Version 8.1 from a later version of the DB2 database system, any registry changes that were made are lost. The registry reverts to the version 8.1 `HealthRules.reg` file that contains the settings that existed before you upgraded and started using the settings in the newer registry file.

Health indicator data: The health monitor records a set of data for each health indicator on each database partition, including:

- Health indicator name
- Value
- Evaluation timestamp
- Alert state
- Formula, if applicable
- Additional information, if applicable
- History of up to ten of the most recent health indicator evaluations. Each history entry captures the following health indicator evaluations leading up to the current health indicator output:
  - Value
  - Formula (if applicable)
  - Alert state
  - Timestamp

The health monitor also tracks the highest severity alert state at the instance, database, and table space levels. At each level, this health indicator represents the highest severity alert existing for health indicators at that level, or any of the levels below it. For example, the highest severity alert state for an instance includes health indicators on the instance, any of its database, and any of the table spaces and table space containers for each of the databases.

Capturing database health snapshots:

Capturing a database health snapshot using SQL table functions:

You can capture database health snapshots using SQL table functions. Each available health snapshot table function corresponds to a health snapshot request type.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

About this task

To capture a database health snapshots using SQL table functions:

Procedure

1. Identify the SQL table function you plan to use.

   SQL table functions have two input parameters:

   - A VARCHAR(255) for the database name
   - An INT for the partition number (a value between 0 and 999). Enter the integer corresponding to the partition number you want to monitor. To capture a snapshot for the currently connected partition, enter a value of -1. To capture a global snapshot, enter a value of -2.

   Note: The database manager snapshot SQL table functions are the only exception to this rule because they have only one parameter. The single parameter is for partition number. If you enter NULL for the database name parameter, the monitor uses the database defined by the connection through which the table function has been called.

2. Issue the SQL statement.

   The following example captures a basic health snapshot for the currently connected partition, and on the database defined by the connection from which this table function call is made:

   ```
   SELECT * FROM TABLE( HEALTH_DB_INFO( cast (NULL as VARCHAR(1)), -1))
          as HEALTH_DB_INFO
   ```

   You can also select individual monitor elements from the returned table. Each column in the returned table corresponds to a monitor element. Accordingly, the monitor element column names correspond directly to the monitor element names. The following statement returns only the db path and server platform monitor elements:

   ```
   SELECT db_path, server_platform
          FROM TABLE( HEALTH_DB_INFO( cast (NULL as VARCHAR(1)), -1 ) )
          as HEALTH_DB_INFO
   ```

Capturing a database health snapshot using the CLP:

You can capture health snapshots using the `GET HEALTH SNAPSHOT` command from the CLP. The command syntax supports retrieval of health snapshot information for the different object types monitored by the health monitor.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Before you begin

You must have an instance attachment to capture a health snapshot. If there is not an attachment to an instance, then a default instance attachment is created. To obtain a snapshot of a remote instance, you must first attach to that instance.

About this task

To capture a database health snapshot using the CLP

Procedure

1. From the CLP, issue the GET HEALTH SNAPSHOT command with the desired parameters.

   In the following example, a database manager level health snapshot is captured immediately after starting the database manager.

   ```
   db2 get health snapshot for dbm
   ```

2. For partitioned database systems, you can capture a database snapshot specifically for a certain partition or globally for all partitions. To capture a health snapshot for a database on a specific partition (for example, partition number 2), issue the following command:

   ```
   db2 get health snapshot for db on sample at dbpartitionnum 2
   ```

   To capture a database snapshot for all applications on all partitions, issue the following command:

   ```
   db2 get health snapshot for db on sample global
   ```

   The following command captures a health snapshot with additional detail, including the formula, additional information, and health indicator history:

   ```
   db2 get health snapshot for db on sample show detail
   ```

3. For collection state-based health indicators, you can capture a database snapshot for all collection objects, regardless of state. The regular GET HEALTH SNAPSHOT FOR DB command returns all collection objects requiring an alert for all collection state-based health indicators.

   To capture a health snapshot for a database with all collection objects listed, issue the following command:

   ```
   db2 get health snapshot for db on sample with full collection
   ```

Capturing a database health snapshot from a client application:

You can capture health snapshots using the snapshot monitor API in a C or C++ application. A number of different health snapshot request types can be accessed by specifying parameters in the db2GetSnapshot API.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Before you begin

You must be attached to an instance to capture a health snapshot. If there is not an attachment to an instance, then a default instance attachment is created. To obtain a snapshot of a remote instance, you must first attach to that instance.

Procedure

1. Include the sqlmon.h and db2ApiDf.h DB2 libraries in your code. These libraries are found in the `sqllib\include` directory.

   ```
   #include <db2ApiDf.h>
   #include <sqlmon.h>
   ```

2. Set the snapshot buffer unit size to 50 KB.

```
#define SNAPSHOT_BUFFER_UNIT_SZ 51200
```

3. Declare the sqlma, sqlca, sqlm_collected, and db2GetSnapshotData structures.

```
struct sqlma *pRequestedDataGroups;
struct sqlca sqlca;
memset (&sqlca, '\0', sizeof(struct sqlca));
struct sqlm_collected collectedData;
memset (&sqlm_collected, '\0', sizeof(struct sqlm_collected));
db2GetSnapshotData getSnapshotParam;
memset(&db2GetSnapshotData, '\0', sizeof(db2GetSnapshotData));
```

4. Initialize a pointer to contain the snapshot buffer, and to establish the buffer's size.

```
static sqluint32 snapshotBufferSize = SNAPSHOT_BUFFER_UNIT_SZ;
sqluint32 outputFormat;
char *snapshotBuffer;
```

5. Initialize the sqlma structure and specify that the snapshot you are capturing is of database manager level information.

```
pRequestedDataGroups = (struct sqlma *)malloc(SQLMASIZE(1));
memset(&pRequestedDataGroups, '\0', sizeof(struct pRequestedDataGroups));
pRequestedDataGroups->obj_num = 1;
pRequestedDataGroups->obj_var[0].obj_type = SQLMA_DB2;
```

6. Initialize the buffer, which will hold the snapshot output.

```
snapshotBuffer = (char *)malloc(snapshotBufferSize);
memset (&snapshotBuffer, '\0', sizeof(snapshotBuffer));
```

7. Populate the db2GetSnapshotData structure with the snapshot request type (from the sqlma structure), buffer information, and other information required to capture a snapshot.

```
getSnapshotParam.piSqlmaData = pRequestedDataGroups;
getSnapshotParam.poCollectedData = &collectedData;
getSnapshotParam.poBuffer = snapshotBuffer;
getSnapshotParam.iVersion = SQLM_DBMON_VERSION9_5;
getSnapshotParam.iBufferSize = snapshotBufferSize;
getSnapshotParam.iStoreResult = 0;
getSnapshotParam.iNodeNumber = SQLM_CURRENT_NODE;
getSnapshotParam.poOutputFormat = &outputFormat;
getSnapshotParam.iSnapshotClass = SQLM_CLASS_HEALTH;
```

8. Capture the health snapshot. Pass the following parameters:
   - db2GetSnapshotData structure, which contains the information necessary to capture a snapshot
   - A reference to the buffer where snapshot output is directed.

```
db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
```

9. Include logic to handle buffer overflow. After a snapshot is taken, the sqlcode is checked for a buffer overflow. If a buffer overflow occurs, the buffer is cleared, reinitialized, and the snapshot is taken again.

```
while (sqlca.sqlcode == 1606)
{
  free(snapshotBuffer);
  snapshotBufferSize += SNAPSHOT_BUFFER_UNIT_SZ;
  snapshotBuffer = (char *)malloc(snapshotBufferSize);
  if (snapshotBuffer == NULL)
  {
    printf("\nMemory allocation error.\n");
    return;
  }
```

```
  getSnapshotParam.iBufferSize = snapshotBufferSize;
  getSnapshotParam.poBuffer = snapshotBuffer;
  db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
}
```

10. Process the snapshot monitor data stream. Refer to the figure following these steps to see the snapshot monitor data stream.
11. Clear the buffer.

```
free(snapshotBuffer);
free(pRequestedDataGroups);
```

Results

After you capture a health snapshot with the db2GetSnapshot API, the API returns the health snapshot output as a self-describing data stream. The following example shows the data stream structure:

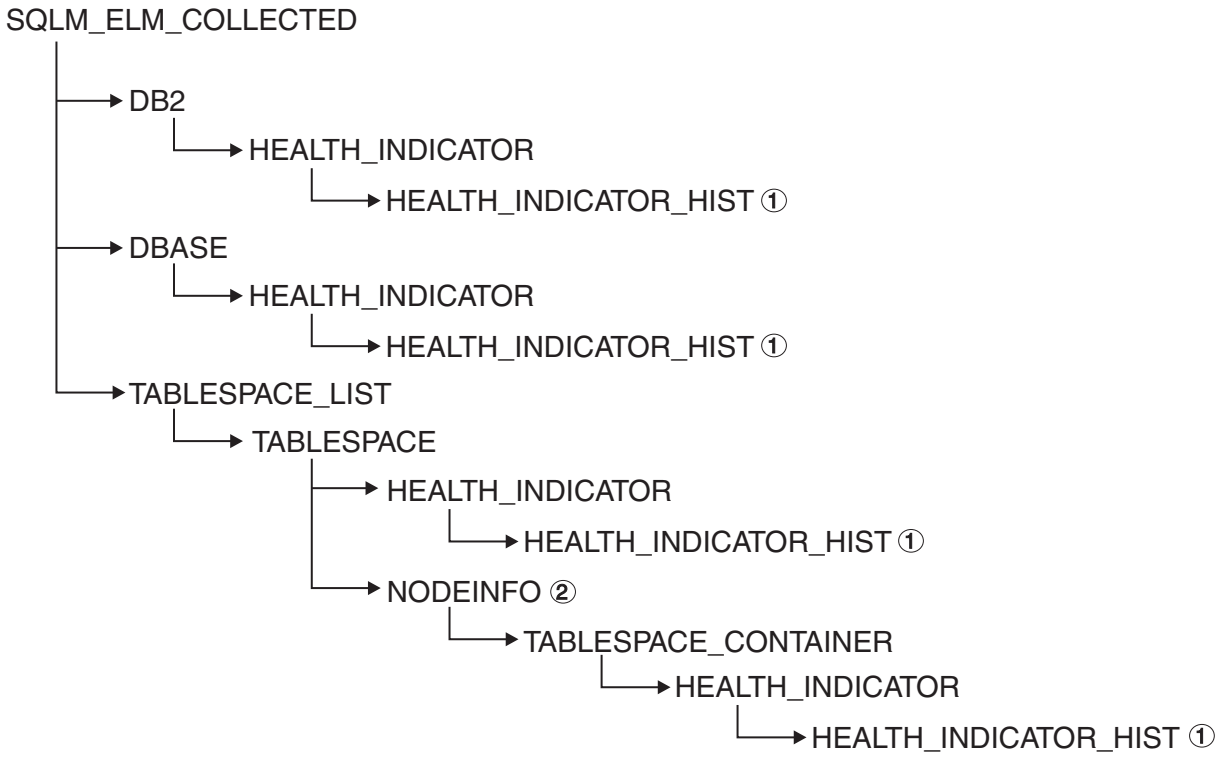


Figure 16. Health snapshot self-describing data stream

Legend:

1. Only available when the SQLM_CLASS_HEALTH_WITH_DETAIL snapshot class is used.
2. Only available in DB2 Enterprise Server Edition. Otherwise, table space container stream follows.

The following hierarchies display the specific elements in the health snapshot self-describing data stream.

The hierarchy of elements under SQLM_ELM_HI:

```
SQLM_ELM_HI
   SQLM_ELM_HI_ID
   SQLM_ELM_HI_VALUE
```

```
      SQLM_ELM_HI_TIMESTAMP
         SQLM_ELM_SECONDS
         SQLM_ELM_MICROSEC
   SQLM_ELM_HI_ALERT_STATE
```

The hierarchy of elements under SQLM_ELM_HI_HIST, available only with the SQLM_CLASS_HEALTH_WITH_DETAIL snapshot class:

```
SQLM_ELM_HI_HIST
   SQLM_ELM_HI_FORMULA
   SQLM_ELM_HI_ADDITIONAL_INFO
      SQLM_ELM_HEALTH_INDICATOR_HIST
         SQLM_ELM_HI_ID
         SQLM_ELM_HI_VALUE
            SQLM_ELM_HI_TIMESTAMP
               SQLM_ELM_SECONDS
               SQLM_ELM_MICROSEC
         SQLM_ELM_HI_ALERT_STATE
         SQLM_ELM_HI_FORMULA
         SQLM_ELM_HI_ADDITIONAL_INFO
```

The hierarchy of elements under SQLM_ELM_OBJ_LIST:

```
SQLM_ELM_HI_OBJ_LIST
   SQLM_ELM_HI_OBJ_NAME
   SQLM_ELM_HI_OBJ_DETAIL
   SQLM_ELM_HI_OBJ_STATE
      SQLM_ELM_HI_TIMESTAMP
         SQLM_ELM_SECONDS
         SQLM_ELM_MICROSEC
```

The hierarchy of elements under SQLM_ELM_OBJ_LIST_HIST, available only with the SQLM_CLASS_HEALTH_WITH_DETAIL snapshot class:

```
SQLM_ELM_HI_OBJ_LIST_HIST
   SQLM_ELM_HI_OBJ_NAME
   SQLM_ELM_HI_OBJ_STATE
      SQLM_ELM_HI_TIMESTAMP
         SQLM_ELM_SECONDS
         SQLM_ELM_MICROSEC
```

Health monitor sample output:

The following examples show health snapshots taken using the CLP, and their corresponding output, and illustrate the nature of the health monitor. The objective in the examples is to check the overall health status immediately after starting the database manager.

1. Take the database manager snapshot, using the GET HEALTH SNAPSHOT command:

   ```
   db2 get health snapshot for dbm
   ```

   After the GET HEALTH SNAPSHOT command is issued from the CLP, the snapshot output is directed to the screen.

   ```
   Node name                                    =
   Node type                                    = Database Server with local
                                                  and remote clients
   Instance name                                = DB2
   Snapshot timestamp                           = 11-07-2002 12:43:23.613425

   Number of database partitions in DB2 instance  = 1
   Start Database Manager timestamp             = 11-07-2002 12:43:18.000108
   Instance highest severity alert state        = Not yet evaluated
   ```

```
Health Indicators:

    Not yet evaluated
```

2. Analyze the output. From this health snapshot, you can see that the instance highest severity alert state is "Not yet evaluated". The instance is in this state because the health monitor has just started and has not yet evaluated any health indicators.

   Should the instance highest severity alert state not change:

   - Check the value of the HEALTH_MON database manager configuration parameter to determine if the health monitor is on.
   - If `HEALTH_MON=OFF`, then the health monitor is not started. To start the health monitor, issue the `UPDATE DBM CFG USING HEALTH_MON ON` command.
   - If `HEALTH_MON=ON`, attach to the instance to activate the health monitor. If an instance attachment exists, it is possible that the health monitor could not be loaded into memory.

Another example of taking a database health snapshot using the CLP is outlined below.

1. Before you begin, ensure that a database connection exists, and that the database is quiesced.
2. Take the database manager snapshot, using the GET HEALTH SNAPSHOT command:

   ```
   db2 get health snapshot for db on sample
   ```

3. After the GET HEALTH SNAPSHOT command is issued from the CLP, the snapshot output is directed to the screen.

```
             Database Health Snapshot

Snapshot timestamp                          = 12-09-2002 11:44:37.793184

Database name                               = SAMPLE
Database path                               = E:\DB2\NODE0000\SQL00002\
Input database alias                        = SAMPLE
Operating system running at database server= NT
Location of the database                    = Local
Database highest severity alert state       = Attention

Health Indicators:

 ...
    Indicator Name                            = db.log_util
       Value                                  = 60
       Unit                                   = %
       Evaluation timestamp                   = 12-09-2002 11:44:00.095000
       Alert state                            = Normal

    Indicator Name                            = db.db_op_status
       Value                                  = 2
       Evaluation timestamp                   = 12-09-2002 11:44:00.095000
       Alert state                            = Attention
```

4. Analyze the output.

   This health snapshot reveals that there is an attention alert on the *db.db_op_status* health indicator. The value of 2 indicates that the database is in quiesced state.

Global health snapshots:

On a partitioned database system you can take a health snapshot of the current partition, a specified partition, or all partitions. When taking a global health snapshot across all the partitions of a partitioned database, data is aggregated, where possible, before the results are returned.

The aggregated alert state for the health indicator is equivalent to the highest severity alert state across all the database partitions. Additional information and history data cannot be aggregated across the database partitions, and therefore are not available. The remaining data for the health indicator is aggregated as detailed in the table below.

Table 34. Aggregation of health indicator value, timestamp, and formula data

Health indicator	Aggregation details
• db2.db2_op_status • db2.sort_privmem_util • db2.mon_heap_util • db.db_op_status • db.sort_shrmem_util • db.spilled_sorts • db.log_util • db.log_fs_util • db.locklist_util • db.apps_waiting_locks • db.db_heap_util • db.db_backup_req • ts.ts_util	The health indicator value is obtained from the partition that contains the highest value. The evaluation timestamp and formula are obtained from the same partition.
• db.max_sort_shrmem_util • db.pkgcache_hitratio • db.catcache_hitratio • db.shrworkspace_hitratio	The health indicator value is obtained from the partition that contains the lowest value. The evaluation timestamp and formula are obtained from the same partition.
• db.deadlock_rate • db.lock_escal_rate	The health indicator value is the sum of the values across all the database partitions. The evaluation timestamp and formula cannot be aggregated and are not available.
• ts.ts_op_status • tsc.tscont_op_status • tsc.tscont_util	These health indicators is not aggregated.
• db.hadr_op_status • db.hadr_log_delay	These health indicators are not supported in a multiple partition database.
• db.tb_reorg_req • db.tb_runstats_req • db.fed_nicknames_op_status • db.fed_servers_op_status	This health indicator is evaluated only on one partition, so no aggregation is required. The data is returned from the partition which is evaluating the health indicator.

Note: When taking a global snapshot on a single partition object, the output includes all the attributes because there are no partitions to aggregate.

Graphical tools for the health monitor:

Health Center

The Health Center is a graphical administration tool designed to support management-by-exception. For all Windows, Linux, and UNIX instances and databases cataloged on the client, the Health Center provides:

- A central location to view the rolled up alert state of all instances and their databases
- A graphical interface to view current alerts on the instances and databases and their children objects
- A graphical interface to access details and recommended resolution actions for current alerts

To start the Health Center from the command line, type the `db2hc` command.

Important: The Health Center has been deprecated in Version 9.7 and might be removed in a future release. Start using IBM Data Studio and IBM Optim tools. For a mapping between these recommended tools and Control Center tools, see "Table of recommended tools versus Control Center tools" in the *What's New for DB2 Version 9.7* book.

On Windows, you can also start the Health Center from the Start Menu by clicking **Start → Programs → IBM DB2 →<DB2 copy name>→ Monitoring Tools → Health Center**.

The Health Center has a navigation tree in the left panel and an Alerts view in the right panel. The contents of the navigation view are filtered based on the toggle button selected at the top of the navigation view.

The Health Center opens with the **Object in Any Alert State** toggle button selected, which helps to identify those instances with current alerts that should be addressed. When the **All Objects** toggle button is selected, all Windows, Linux, and UNIX instances cataloged on the client and their respective states are displayed. Instances without an icon do not have the health monitor running or are instances prior to version 8, which lack support for health monitor functionality.

When you select an instance, the Health Center requests status from the health monitor for the selected instance. The Alerts view fills with all current alerts for the instance, any of its databases, and any of the table spaces and table space containers of each database. If you expand the instance in the navigation view and select a child database object, the Alerts view is restricted to alerts for the selected database and any of its table spaces or table space containers.

The refresh icon is located in the upper-right corner of the Health Center. Clicking the refresh icon for immediate refresh, or setting a particular refresh interval, causes the Health Center to query the health monitor on the server for its current status. This query does not cause the health monitor to refresh the health indicator evaluations. Each health indicator has a defined refresh interval. Only when the refresh interval has passed will the health indicator be reevaluated for alert state. Only the current status of the health indicators is shown on each timed refresh or requested refresh of the Health Center.

The Alerts view has a function to define customized views with specific customized columns and sorting orders. There are six predefined views in the Health Center that you can customize to your personal naming and categorization

scheme. You can select the predefined views by using the toolbar at the bottom of the window or by selecting **Saved Views** in the **View** menu. To define your own customized views, click the **View** button on the toolbar at the bottom of the window, or use the **View** menu. The view that is selected for displaying data in the Alerts view is remembered on the next invocation of the Health Center.

To get the details for an alert, select the alert row in the Alerts view. Using the **Selected** menu, or by right-clicking the row, select **Show Details**. The Details window shows the detailed information for the alert including the object and partition where the alert occurred, the formula (if applicable), and value for the health indicator.

For threshold-based health indicators, the thresholds that were used in determining the alert condition are displayed. The Details window also displays additional information for the health indicator. This information might include values for configuration parameters or other monitor data that provides context for the alert. A description of the health indicator is displayed, including the purpose for the health indicator and why it is an important attribute to measure.

For collection state-based health indicators, the list of collection objects is displayed in the Objects in **Health Indicator Alert State** table. Object name, state, timestamp, and details are provided in the table.

A **View History** button is provided on the details page. History records are stored for the health indicator starting with the second refresh of the health indicator evaluation. Content is displayed in the View History dialog in the Health Center only after the history records are stored. The history of collection objects, for collection state-based health indicators, can be viewed by clicking the **View Collection History** button in the History window.

Health Center Status Beacon

The Health Center Status Beacon is a visual indicator that can be enabled in the DB2 administration tools. When the Health Center is not open, the beacon will notify you of current alerts while you are working with other DB2 administration tools. The beacon is intended to prompt the user to open the Health Center because of an alert condition.

The Health Center Status Beacon has two different notification methods. One notification method uses a pop-up message. Another notification method uses a graphical beacon that displays on the right portion of the status line of open windows. The graphical beacon includes a button that provides single-click access to the Health Center.

Both beacon notification methods are enabled through the Tools Settings dialog. The "notify through pop-up" method controls the pop-up message notification, and the "notify through status line" method controls the visual beacon.

Retrieving health recommendations:

Health recommendation queries with SQL:

Recommendations can be queried with SQL using the SYSPROC.HEALTH_HI_REC stored procedure.

When using the SYSPROC.HEALTH_HI_REC stored procedure, recommendations are returned in an XML document that is:

- Formatted according to the health recommendations XML schema DB2RecommendationSchema.xsd located in the `sqllib\misc` directory.
- Encoded in UTF-8 and contains text in the client language.
- Organized as a collection of recommendation sets, where each recommendation set describes a problem (health indicator) being resolved and contains one or more recommendations to resolve that health indicator. Refer to the schema definition for specific details about information that can be retrieved from the document.

All information available through the CLP is also available in the XML recommendation document that is returned when you query with SQL.

The SYSPROC.HEALTH_HI_REC stored procedure takes the following arguments:

- A health indicator
- A definition of the object on which the health indicator has entered an alert state

The output recommendation document is returned as a BLOB. Therefore, it is not helpful to work with this stored procedure from the command line, since the CLP will limit the amount of output displayed. It is recommended that this stored procedure be invoked using a high level language (such as C or Java) that allows the returned XML document to be properly parsed to retrieve any required elements and attributes.

Retrieving health recommendations using the CLP:

Recommendations can be retrieved using the GET RECOMMENDATIONS command from the CLP. The command syntax supports querying recommendations to resolve a specific health alert, such as a health indicator that has currently entered an alert state on a particular object.

Before you begin

You must have an instance attachment to retrieve recommendations from the health monitor. If there is not an attachment to an instance, a default instance attachment is created. To obtain recommendations from a health monitor on a remote instance, you must first attach to that instance. No special authority is required to retrieve recommendations from the health monitor.

About this task

The command syntax also supports retrieval of the complete set of recommendations for a given health indicator, which does not have to be in an alert state when the command is executed. Recommendations for resolving an alert on a specific health indicator can be queried at either a single partition level or a global level.

When querying recommendations on a health alert on a specific object, the health monitor is solving a specific alert and is able to provide details on the alert being resolved in the problem section of the output.

The health monitor is also able to provide a ranking for the recommendations and, in some cases, it might be able to generate scripts that can be executed to resolve the alert. Additionally, the health monitor might reject and not display some recommendations if they are not applicable to the particular problem situation. On

the other hand, if recommendations are queried by health indicator name only, as in the first example below, the total set of possible recommendations will always be returned. In such cases, the CLP command is simply providing information about actions that a user should consider undertaking if they see an alert.

Procedure

Retrieve the recommendations using the GET RECOMMENDATIONS command:

1. You might want to issue the following command to see the total set of actions that could be recommended to resolve an alert on the **db.db_op_status** health indicator.

   ```
   db2 get recommendations for health indicator db.db_op_status
   ```

   In this example, the full set of recommendations for the **db.db_op_status** health indicator is returned. The health indicator does not have to be in an alert state to issue this command.

   This output shows that there are two possible recommendations for this health indicator: unquiesce the database or investigate rollforward progress on the database. Because the command is being used to query all possible recommendations, rather than to ask how to resolve a specific alert, the health monitor cannot identify the best recommendation in this case. As a result, the full set of recommendations is returned.

   ```
   Recommendations:

   Recommendation: Investigate rollforward progress.

   A rollforward is in progress on the database due to an explicit
   request from the administrator. You have to wait for the rollforward
   to complete for the instance to return to active state.

   Take one of the following actions:

   Launch DB2 tool: Utility Status Manager

   The Utility Status Manager allows you to monitor the progress
   and change the priority of currently running utilities.

   To open the Utility Status Manager:
   1.  From the Control Center, expand the object tree until you find
       the database that you want.
   2.  Right-click the database, and click Manage Utilities from the
       pop-up menu. The Utility Status Manager opens.

   To view progress of the rollforward utility, right-click on the
   rollforward utility and select View Progress Details.


   From the Command Line Processor, issue the commands shown in the
   following example to view the progress of the rollforward
   utility:

   LIST UTILITIES SHOW DETAIL


   Recommendation: Unquiesce the database.

   The database has been put into QUIESCE PENDING or QUIESCE state by
   an explicit request from the administrator. If you have QUIESCE_CONNECT
   authority, or are DBADM or SYSADM, you will still have access to the
   database and will be able to use it normally. For all other users, new
   connections to the database are not permitted and new units of work
   cannot be started. Also, depending on the quiesce request, active
   ```

```
units of work will be allowed to complete or will be rolled back
immediately. You can issue an unquiesce to return to active state.

Take one of the following actions:

Launch DB2 tool: Control Center Unquiesce Database

The Control Center has an option on a database that can be used to
unquiesce the database.

To unquiesce a database:
1.  From the Control Center, expand the object tree until you find
    the database that you want.
2.  Right-click the database, and click Unquiesce from the pop-up
    menu. The database is unquiesced.


From the Command Line Processor, issue the commands shown in the
following example:

CONNECT TO DATABASE database-alias
UNQUIESCE DATABASE
```

2. Suppose you observe that the health indicator **db.db_heap_util** has entered an alert state for the database SAMPLE, and you want to determine how to resolve the alert. In this case, you want to resolve a specific problem, therefore you could issue the GET RECOMMENDATIONS command in the following way:

   ```
   db2 get recommendations for health indicator db.db_heap_util
       for database on sample
   ```

   This output shows a summary of the problem and a set of recommendations to resolve the problem. The health monitor has ranked the recommendations in its order of preference. Each recommendation contains a description and a set of actions that indicate how to perform the recommended action.

   ```
   Problem:

       Indicator Name                          = db.db_heap_util
          Value                                = 42
          Evaluation timestamp                 = 11/25/2003 19:04:54
          Alert state                          = Alarm
          Additional information               =

   Recommendations:

   Recommendation: Increase the database heap size.
   Rank: 1

   Increase the database configuration parameter dbheap sufficiently
   to move utilization to normal operating levels. To increase the
   value, set the new value of dbheap to be equal to
   (pool_cur_size / (4096*U)) where U is the desired utilization rate.
   For example, if your desired utilization rate is 60% of the warning
   threshold level, which you have set at 75%, then
   U = 0.6 * 0.75 = 0.45 (or 45%).

   Take one of the following actions:

   Execute the following scripts at the DB2 server (this can be done using
   the EXEC_DB2_CMD stored procedure):

   CONNECT TO DATABASE SAMPLE;
   UPDATE DB CFG USING DBHEAP 149333;
   CONNECT_RESET;
   ```

```
Launch DB2 tool: Database Configuration Window

The Database Configuration window can be used to view and update
database configuration parameters.

To open the Database Configuration window:
1.  From the Control Center, expand the object tree until you find
    the databases folder.
2.  Click the databases folder. Any existing database are displayed
    in the contents pane on the right side of the window.
3.  Right-click the database that you want in the contents pane, and
    click Configure Parameters in the pop-up menu. The Database
    Configuration window opens.

On the Performance tab, update the database heap size parameter as
suggested and click OK to apply the update.


Recommendation: Investigate memory usage of database heap.
Rank: 2

There is one database heap per database and the database manager uses
it on behalf of all applications connected to the database. The data
area is expanded as needed up to the maximum specified by dbheap.

For more information on the database heap, refer to the DB2
Information Center.

Investigate the amount of memory that was used for the database heap
over time to determine the most appropriate value for the database
heap configuration parameter. The database system monitor tracks the
highest amount of memory that was used for the database heap.

Take one of the following actions:

Launch DB2 tool: Memory Visualizer

The Memory Visualizer is used to monitor memory allocation within
a DB2 instance. It can be used to monitor overall memory usage, and
to update configuration parameters for individual memory components.

To open the Memory Visualizer:
1.  From the Control Center, expand the object tree until you find
    the instances folder.
2.  Click the instances folder. Any existing instances are displayed
    in the contents pane on the right side of the window.
3.  Right-click the instance that you want in the contents pane, and
    click View Memory Usage in the pop-up menu. The Memory Visualizer
    opens.

To start the Memory Visualizer from the command line issue the
db2memvis command.

The Memory Visualizer displays a hierarchical list of memory pools
for the database manager. Database Heap is listed under the Database
Manager Memory group for each database. On Windows, it is listed
under the Database Manager Shared Memory group.

Click the check box on the Show Plot column for the Database Heap
row to add the element to the plot.
```

3. For partitioned database systems you can query recommendations for a health indicator that has entered an alert state on a certain partition, or globally for all partitions. When recommendations are queried globally, a set of recommendations is returned that applies to the health indicator on all

partitions. For example, if the health indicator is in an alert state on partitions 1 and 3, a collection of two scripts might be returned where each script is to be applied to a different partition.

The following example shows how to query recommendations for a health indicator on a specific partition (in this example, partition number 2):

```
db2 get recommendations for health indicator db.db_heap_util
  for database on sample at dbpartitionnum 2
```

The following example shows how to retrieve a set of recommendations to resolve a health indicator that is in an alert state on several partitions:

```
db2 get recommendations for health indicator db.db_heap_util
  for database on sample global
```

Retrieving health recommendations using a client application:

Recommendations can be queried using the db2GetRecommendations API in a C or C++ application.

Before you begin

You must have an instance attachment to capture a health snapshot. If there is not an attachment to an instance, then a default instance attachment is created. To query recommendations on a remote instance, you must first attach to that instance.

About this task

When using the db2GetRecommendations API, recommendations are returned in an XML document that is:

- Formatted according to the health recommendations XML schema DB2RecommendationSchema.xsd located in the `MISC` subdirectory within the `SQLLIB` directory.
- Encoded in UTF-8 and contains text in the client language.
- Organized as a collection of recommendation sets, where each recommendation set describes a problem (health indicator) being resolved and contains one or more recommendations to resolve that health indicator. Refer to the schema definition for specific details about what information that can be retrieved from the document.

All information available through the CLP is also available in the XML recommendation document that is returned.

To retrieve health recommendations using a client application:

Procedure

1. Include the sqlmon.h and db2ApiDf.h DB2 header files. These are found in the `sqllib\include` directory.

   ```
   #include <db2ApiDf.h>
   #include <sqlmon.h>
   ```

2. Declare the sqlca, and the db2GetRecommendationsData structure.

   ```
   struct sqlca sqlca ;
   db2GetRecommendationsData recData ;

   memset( &sqlca, '\0', sizeof( struct sqlca ) ) ;
   memset( &recData, '\0', sizeof( db2GetRecommendationsData ) ) ;
   ```

3. Populate the db2GetRecommendationsData structure with information about the alert for which you want to retrieve recommendations. In the code excerpt that follows, recommendations are being queried for the **db2.db_heap_util** health indicator on the Sample database.

```
recData.iSchemaVersion = DB2HEALTH_RECSCHEMA_VERSION8_2 ;
recData.iNodeNumber = SQLM_CURRENT_NODE ;
recData.iIndicatorID = SQLM_HI_DATABASE_HEAP_UTILIZATION  ;
recData.iObjType = DB2HEALTH_OBJTYPE_DATABASE ;
recData.piDbName = "SAMPLE" ;
```

4. Invoke the db2GetRecommendations API to retrieve recommendations for an alert on this health indicator on the specified database.

```
db2GetRecommendations(  db2Version820, &recData, &sqlca ) ;
```

5. Check the sqlcode returned in the sqlca for any errors that occurred. If the API call was successful, process the recommendation XML document that is returned in the poRecommendation field of the db2GetRecommendationsData structure. Use your choice of XML parser to extract the required elements or attributes. Refer to the DB2RecommendationSchema.xsd XML schema in the `sqllib\misc` directory for details about the information that can be retrieved from the XML document.
6. Free any memory allocated by the db2GetRecommendations API. This will free the recommendation document returned in the poRecommendation field of the db2GetRecommendationsData structure.

```
db2GetRecommendationsFree( db2Version820, &recData, &sqlca );
```

Results

Typically you would combine the preceding code with a call to the snapshot APIs to take a health snapshot because recommendations are generally queried when you detect a health indicator has entered an alert state.

Resolving health monitor alerts using the Health Center:

The Health Center provides support to retrieve and implement recommended actions for alert conditions.

About this task

To resolve health monitor alerts using the Health Center:

Procedure

1. From the Health Center alerts view, right-click the row of the alert that you want to resolve and select **Recommendation Advisor** from the pop-up menu. The Recommendation Advisor opens and displays the details of the alert in a format similar to the Details window.
2. Follow the steps of the Recommendation Advisor to select the most appropriate recommendation. The Recommendation Advisor provides the functionality to implement the recommendation.

Results

There are two types of recommendations: investigation and recommendation. The following four types of actions are supported in the Recommendation Advisor for these recommendation types:

Launching a graphical administration tool
This option will launch a graphical tool that will resolve or investigate the alert condition. The tool is launched in the context of the object against which the alert occurred.

Updating configuration parameters
The configuration parameters requiring updates are listed with current and suggested values. The suggested value can be updated as needed.

Running a DB2 command script
The recommendation action might require more than a single command. DB2 command scripts allow for multiple commands to be run to resolve the alert condition. For example, the Reorganization Required health indicator provides a DB2 command script action to run the utility.

Implementing an alternative resolution
If the action cannot be accomplished within the DB2 administration toolset, instructions are provided to resolve the alert condition using alternate methods.

Health indicator configuration:

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

A default health monitor configuration is provided during installation. This ensures that the health monitor can evaluate the health of the database environment as soon as DB2 is started. However, the health monitor's behavior in evaluating health indicators and reacting to alert states can be fine-tuned through configuration for a specific user's environment.

There are different levels at which the configuration can be defined. A default configuration of factory settings is provided for each health indicator when DB2 is installed. When the health monitor starts for the first time, a copy of the factory settings provides the defaults for the instance and global settings.

Instance settings apply to the instance. Global settings apply to objects such as databases, table spaces, and table space containers in the instance that do not have customized settings defined.

Updating health indicator settings for a specific database, table space, or table space container creates object settings for the updated health indicators. The default for object settings is the global settings.

The health monitor checks the object settings when it processes a health indicator for a particular database, table space, or table space container. If the settings for a particular health indicator have never been updated, the default global settings are used to process the health indicator. The instance settings are used when the health monitor processes a health indicator for the instance.

You can alter health monitor behavior by using a number of attributes that can be configured for each health indicator. The first set of parameters (evaluation flag, thresholds, sensitivity) defines when the health monitor will generate an alert for a health indicator. The second set of parameters (action flag, actions) defines what the health monitor does upon generating the alert.

Evaluation flag
Each health indicator has an evaluation flag to enable or disable evaluation of alert state.

Warning and alarm thresholds
Threshold-based health indicators have settings defining the warning and alarm regions for the health indicator value. These warning and alarm threshold values can be modified for your particular database environment.

Sensitivity parameter
The sensitivity parameter defines the minimum amount of time, in seconds, that the health indicator value has to be in an alert state before the alert is generated. The wait time associated with the sensitivity value starts on the first refresh interval during which the health indicator value enters an alert state. You can use this value to eliminate erroneous alerts generated due to temporary spikes in resource usage.

Consider an example using the Log Utilization (*db.log_util*) health indicator. Suppose that you review the DB2 notify log on a weekly basis. In the first week, an entry for *db.log_util* is in alarm state. You recall having received notification for this situation, but on checking for the alert situation from the CLP, the health indicator was back in normal state. After a second week, you notice a second alarm notification entry for the same health indicator at the same time of the week. You investigate activity in your database environment on the two occasions that alerts were generated, and you discover that an application that takes a long time to commit is run weekly. This application causes the log utilization to spike for a short time, approximately eight to nine minutes, until the application commits. You can see from the history entries in the alarm notification record in the notification log, that the *db.log_util* health indicator is evaluated every 10 minutes. Because the alert is being generated, the application time must be spanning that refresh interval. You set the sensitivity for the *db.log_util* parameter to ten minutes. Now every time the value of *db.log_util* first enters the warning or alarm threshold regions, the value must remain in that region for at least ten minutes before the alert is generated. No further notification entries are recorded in the notification log for this situation because the application only lasts eight to nine minutes.

Action flag
The running of actions on alert generation is controlled by the action flag. Only when the action flag is enabled will configured alert actions be run.

Actions
Script or task actions can be configured to run on alert occurrence. For threshold-based health indicators, actions can be configured to run on warning or alarm thresholds. For state-based health indicators, actions can be configured to run on any of the possible non-normal conditions. The DB2 administration server must be running for actions to run.

The following input parameters are passed to every operating system command script:

- <health indicator shortname>
- <object name>
- <value | state>
- <alert type>

Script actions use the default interpreter on the operating system. If you want to use a non-default interpreter, create a task in the Task Center with the script content. In a multipartitioned environment, the script defined in the script action must be accessible by all partitions.

The refresh interval at which the health monitor checks each health indicator cannot be configured. The recommendation actions considered by the health monitor cannot be configured.

The health monitor configuration is stored in a binary file, HealthRules.reg:

- On Windows, HealthRules.reg is stored in `x:\<SQLLIB_PATH>\<INSTANCE_NAME>`. For example, `d:\sqllib\DB2`.
- On UNIX, HealthRules.reg is stored in `~/<SQLLIB_PATH>/cfg`. For example, `~/home/sqllib/cfg`.

It is possible to replicate a health monitor configuration to other DB2 Version 8 instances on a Linux, UNIX, or Windows server. You can accomplish this replication by copying the binary configuration file to the appropriate directory location on the target instance.

Retrieving health indicator configuration using the CLP:

The GET ALERT CONFIGURATION command allows you to view the factory settings and the instance, global, and object settings.

Procedure

1. To view the global settings for database-level health indicators, which apply to all databases without customized settings for the health indicator, issue the following command:

   ```
   DB2 GET ALERT CONFIGURATION FOR DATABASES
   ```

2. To view the global settings for database-level health indicators, which apply to all databases without customized settings for the health indicator, issue the following command:

   ```
   DB2 GET ALERT CONFIGURATION FOR DATABASES
   ```

   The output of each health indicator's settings indicates whether it has been changed from its default. In the following output, the global settings have not been updated; therefore, they are the same as the default factory settings. To view factory settings for database-level health indicators, issue the same command as in the preceding example with the DEFAULT keyword.

   ```
            Alert Configuration

   Indicator Name                    = db.db_op_status
       Default                       = Yes
       Type                          = State-based
       Sensitivity                   = 0
       Formula                       = db.db_status;
       Actions                       = Disabled
       Threshold or State checking   = Enabled

   Indicator Name                    = db.sort_shrmem_util
       Default                       = Yes
       Type                          = Threshold-based
       Warning                       = 70
       Alarm                         = 85
       Unit                          = %
       Sensitivity                   = 0
       Formula                       = ((db.sort_shrheap_allocated/sheapthres_shr)
                                        *100);
       Actions                       = Disabled
       Threshold or State checking   = Enabled
   ...
   ```

3. To view the custom settings for the SAMPLE database, issue the following command:

   ```
   DB2 GET ALERT CONFIGURATION FOR DATABASE ON SAMPLE
   ```

   If there are no specific settings for a particular health indicator on the object specified, the global settings for all databases are displayed. To view the settings for a particular health indicator, add the `USING health-indicator-name` clause to any of the preceding examples.

Health indicator configuration updates using the CLP: The health indicator configuration for a particular health indicator can be updated for the global settings or the object settings for a particular object.

The UPDATE ALERT CONFIGURATION command has four sub-clauses that cover the different update options. Only one sub-clause can be used in each UPDATE ALERT CONFIGURATION command. To use more than one of the options, multiple UPDATE ALERT CONFIGURATION commands must be issued.

The first sub-clause, `SET parameter-name value`, provides support to update:

- The evaluation flag
- The warning and alarm thresholds (if applicable)
- The sensitivity flag
- The action flag

The parameter names for these settings are, respectively:

- THRESHOLDSCHECKED
- WARNING and ALARM
- SENSITIVITY
- ACTIONSENABLED

The other three sub-clauses provide support to add, to update, and to delete script or task actions.

The following commands update a threshold-based health indicator configuration for the *db.spilled_sorts* health indicator on the SAMPLE database. The update changes the warning threshold to 25, to enable actions, and to add a script action:

```
DB2 UPDATE ALERT CONFIGURATION FOR DATABASE ON SAMPLE USING DB.SPILLED_SORTS
    SET WARNING 25, ACTIONSENABLED YES
DB2 UPDATE ALERT CONFIGURATION FOR DATABASE ON SAMPLE USING DB.SPILLED_SORTS
    ADD ACTION SCRIPT c:\myscript TYPE OS COMMAND LINE PARAMETERS 'space'
    WORKING DIRECTORY c:\ ON ALARM USER dba1 PASSWORD dba1
```

The following commands update a state-based health indicator configuration for the *ts.ts_util* health indicator for the global settings. The update defines an action to run when any table space is in backup pending state.

```
DB2 UPDATE ALERT CONFIGURATION FOR TABLESPACES USING TS.TS_UTIL
    SET ACTIONSENABLED YES
DB2 UPDATE ALERT CONFIGURATION FOR TABLESPACES USING TS.TS_UTIL
    ADD ACTION TASK 0.1 ON ATTENTION 32 ON localhost USER dba1 PASSWORD dba1
```

This update will apply to all table spaces for the instance that do not have customized settings for this health indicator.

When adding actions to a health indication configuration, the options for the `ON condition` clause are based on the type of health indicator:

- For a threshold-based health indicator, WARNING and ALARM are valid conditions.
- For a state-based health indicator, the ON ATTENTION *state* option must be used. A valid numeric state, as defined for the health indicator, should be used. The database manager and database operational state values can be found in sqllib\include\sqlmon.h. The table space and table space container operational values are listed in sqllib\include\sqlutil.h. Note that actions cannot be executed for the database manager down state. Refer to the description of the db2.db2_op_status health indicator for details.

Resetting health indicator configuration using the CLP:

The CLP provides support for the global settings to be reset to the factory settings. The object settings for a particular object can also be reset to the custom settings for that object type.

Procedure

- To reset the object settings for the SAMPLE database to the current global settings for databases:

  ```
  DB2 RESET ALERT CONFIGURATION FOR DATABASE ON SAMPLE
  ```

- Issue the following command to reset the global settings for databases to the factory settings:

  ```
  DB2 RESET ALERT CONFIGURATION FOR DATABASES
  ```

- To reset the configuration for a particular health indicator, add the USING *health-indicator-name* clause to any of the preceding examples.

Configuring health indicators using a client application:

Health monitor configuration is accessible through the db2GetAlertCfg, db2UpdateAlertCfg, and db2ResetAlertCfg APIs in a C or C++ application. Each of these APIs can access the factory, instance, global, and object settings.

Before you begin

You must have an instance attachment to access the health monitor configuration. If there is not an attachment to an instance, then a default instance attachment is created. To access the health monitor configuration of a remote instance, you must first attach to that instance.

About this task

Combinations of the **objType** and **defaultType** parameters in the db2GetAlertCfgData structure allow access to the various levels of health indicator configuration.

Table 35. Settings for objType and defaultType to access configuration levels

Setting	objType and defaultType
Factory settings	objType = DB2ALERTCFG_OBJTYPE_{DBM | DATABASES |TABLESPACES | CONTAINERS} and defaultType = DB2ALERTCFG_DEFAULT

Table 35. Settings for objType and defaultType to access configuration levels (continued)

Setting	objType and defaultType
Global settings	objType = DB2ALERTCFG_OBJTYPE_{DBM | DATABASES | TABLESPACES | CONTAINERS} and defaultType = DB2ALERTCFG_NOT_DEFAULT or objType = DB2ALERTCFG_OBJTYPE_{DATABASE | TABLESPACE | CONTAINER} and defaultType = DB2ALERTCFG_DEFAULT
Object settings	objType = DB2ALERTCFG_OBJTYPE_{DATABASE | TABLESPACE | CONTAINER} and defaultType = DB2ALERTCFG_NOT_DEFAULT

Procedure

1. To get the specific object setting for health indicators on the SAMPLE database:
   a. Include the db2ApiDf.h DB2 header file, found in the `sqllib\include` directory.

      ```
      #include <db2ApiDf.h>
      ```

   b. Declare and initialize the sqlca and db2GetAlertCfgData structures.

      ```
      struct sqlca ca;
      memset (&sqlca, '\0', sizeof(struct sqlca));

      char* objName = NULL;
      char* dbName = "SAMPLE";
      db2Uint32 objType = DB2ALERTCFG_OBJTYPE_DATABASE;
      db2Uint32 defaultType = DB2ALERTCFG_NOT_DEFAULT;

      db2GetAlertCfgData data = {objType, objName, defaultType, dbName, 0, NULL} ;
      ```

   c. Call the db2GetAlertCfg API.

      ```
      rc = db2GetAlertCfg (db2Version810, &data, &ca);
      ```

   d. Process the returned configuration and free the buffer allotted by the API.

      ```
      if (rc >= SQLO_OK) {
        if ((data.ioNumIndicators > 0) && (data.pioIndicators != NULL)) {
          db2GetAlertCfgInd *pIndicators = data.pioIndicators;

          for (db2Uint32 i=0; i  data.ioNumIndicators; i++) {
       //process the entry as necessary using fields defined in db2ApiDf.h
          }
        }

        db2GetAlertCfgFree (db2Version810, &data, &ca);
      }
      ```

2. The following steps detail the procedure to update the alert configuration of the **db.sort_shrmem_util** health indicator for the global settings for database objects, setting warning threshold to 80 and adding task action 1.1:
   a. Include the db2ApiDf.h DB2 header file, found in the `sqllib\include` directory.

      ```
      #include <db2ApiDf.h>
      ```

   b. Declare and initialize the sqlca and db2AlertTaskAction structures.

      ```
      struct sqlca ca;
      memset (&sqlca, '\0', sizeof(struct sqlca));

      db2Uint32 objType = DB2ALERTCFG_OBJTYPE_DATABASES;

      db2Uint32 taskCondition = DB2ALERTCFG_CONDITION_WARNING;
      char* taskname = "1.1";
      char* hostname = NULL;
      char* userid = "nobody";
      ```

```
char* password = "nothing";

db2AlertTaskAction newTask={taskname,taskCondition,userid,password,hostname};
```

c. Declare and initialize the db2UpdateAlertCfgData structure.

```
struct db2UpdateAlertCfgData setData;

setData.iObjType = objType;
setData.piObjName = NULL;
setData.piDbName = NULL;

setData.iIndicatorID = 1002;

setData.iNumIndAttribUpdates = 1;
setData.piIndAttribUpdates[0].iAttribID = DB2ALERTCFG_WARNING;
setData.piIndAttribUpdates[0].piAttribValue == 80;

setData.iNumActionUpdates = 0;
setData.piActionUpdates = NULL;

setData.iNumActionDeletes = 0;
setData.piActionDeletes = NULL;

setData.iNumNewActions = 1;
setData.piNewActions[0].iActionType = DB2ALERTCFG_ACTIONTYPE_TASK;
setData.piNewActions[0].piScriptAttribs = NULL;
setData.piNewActions[0].piTaskAttribs = &newTask;
```

d. Call the db2UpdateAlertCfg API.

```
rc = db2UpdateAlertCfg(db2Version810, &setData, &ca);
```

3. The following steps detail the procedure to RESET the custom settings for the MYTS table space in the SAMPLE database.

a. Include the db2ApiDf.h DB2 header file, found in the `sqllib\include` directory.

```
#include <db2ApiDf.h>
```

b. Declare and initialize the sqlca and db2ResetAlertCfgData structures.

```
struct sqlca ca;
memset (&sqlca, '\0', sizeof(struct sqlca));

char* objName = "MYTS";
char* dbName = "SAMPLE";
db2Uint32 objType = DB2ALERTCFG_OBJTYPE_TABLESPACE;

db2ResetAlertCfgData data = {objType, objName, dbName};
```

c. Call the db2ResetAlertCfg API.

```
rc = db2ResetAlertCfg (db2Version810, &data, &ca);
```

Configuring health indicators using the Health Center:

The Health Center provides graphical interfaces to view, update, and reset health indicator configurations. The configuration for health indicators is stored in the health monitor within the instance.

Before you begin

To define, change, enable or disable threshold or sensitivity settings for a health indicator, and to define, change, enable or disable running tasks or scripts when a health alert occurs for a health indicator, you must have one of the following authorities:

- SYSADM
- SYSMAINT
- SYSCTRL

About this task

You can adjust the health indicator settings for an instance, the global health indicator settings for database objects contained in the instance, and for individual database objects.

Procedure

1. To configure health indicators using the Health Center:
   a. Select the instance whose health indicators you want to configure.
   b. From the **Selected** menu, or from the right-click menu, click **Configure** then **Health Indicator Settings**. The Health Indicator Configuration Launchpad opens.
   c. Each level of configuration settings that can be updated has a button on the launchpad. Select the button for the level of configuration that you want to view, update, or reset. Each button launches a Health Indicator Configuration window at the chosen level of configuration settings.
   d. To update the health indicator settings, select the row of the health indicator in the Current health indicator settings table.
   e. From the **Select** menu, or from the right-click menu, select **Edit**. The Configure Health Indicator notebook opens and displays the following information:
      - A description of the health indicator is provided by clicking **Tell Me More**.
      - The evaluation of the health indicator can be enabled and disabled using the **Evaluate** check box.

        Note: The **Evaluate** flag can also be disabled from the Health Center Alerts View for current alerts through the right-click menu option on a current alert. This option will disable the evaluation of the health indicator on the next refresh of the indicator in the health monitor. When selecting **Disable evaluation** for an alert in the Health Center, the evaluation flag is set to false for the health indicator, but the alert will not be removed from the Alerts view until the following events take place:
        – The health monitor refresh interval for that particular health indicator is reached
        – The health monitor refreshes the health indicator evaluation
        – The Health Center refreshes its view of status
      - On the Alert page, for threshold-based health indicators, the warning and alarm thresholds can be updated. The sensitivity for any health indicator can also be set on this page.
      - On the Actions page, a task or script action can be selected to run when an alert occurs. Actions can be configured to run on warning or alarm conditions for threshold-based health indicators or on any non-normal condition for state-based health indicators. You can enable or disable the execution of actions by selecting or deselecting the **Enable actions** check box. To add, update, or remove task or script actions, use the buttons beside the **Script actions** and **Task actions** tables.
2. To view the factory health indicator settings for the instance:
   a. In the Health Indicator Configuration launchpad, click **Instance Settings**.
   b. In the Instance Health Indicator Configuration window, click **View Default**.
3. To view the global health indicator settings for databases, table spaces, or table space containers:

a. In the Health Indicator Configuration launchpad, click **Global Settings**.
b. Select the object type in the Global Health Indicator Configuration window.
c. To view the factory defaults for these global settings, click **View Default**

4. To view the health indicator settings for a database object:
   a. In the Health Indicator Configuration launchpad, click **Object Settings**.
   b. Select the object in the Object Health Indicator Configuration window.
   c. To view the global default health indicator settings for this object type, click **View Default**.

Results

In each of these windows, to reset the settings of all the displayed health indicators to their defaults, click **Reset to Default**. You can also reset individual health indicators by right-clicking one or more health indicators that you want in the **Current health indicator settings** field and selecting **Reset to Default** from the pop-up menu.

Health monitor alert actions on combined states:

Alert actions are tasks or scripts that are run when a health indicator goes into an alert state.

Starting in DB2 V9.1, the health monitor alert actions defined for the health indicator **ts.ts_op_status** on a single alert state are executed whenever this state is set for the table space, irrespective of the other combined states. This makes it possible to run alert actions on a specific table space state even when it is set in conjunction with other states.

In the following example, an alert action script1 defined on attention state QUIESCED:share will run, even if the table space state is QUIESCED:share and QUIESCE:update at the same time.

```
db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status
    set actionsenabled yes')

db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status
    add action script /home/guest001/script1 type operating system command line
    parameters userParam working directory /home/guest001/
    on attention QUIESCED_SHARE on aix1 user guest001 using passw0rd')
```

In the following example, an alert action defined using a combination of states (QUIESCED:share + QUIESCED:update = 3) is executed if and only if the table space state is both QUIESCED:share and QUIESCED:update.

```
db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status
    set actionsenabled yes')

db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status
    add action script /home/guest001/script1 type operating system command line
    parameters userParam working directory /home/guest001/
    on attention 3 on aix1 user guest001 using passw0rd')
```

Starting in DB2 V9.1, health monitor alert actions defined on an object with the same action attributes (name, working directory, command line parameters, host, user and password) run only once, even if it was defined on multiple alert states.

In the following example, the same action is defined on two different alert states. The action is only executed once for a given table space, even if the table space state is in both QUIESCED:share and QUIESCED:update.

```
db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status
    add action script /home/guest001/script1 type operating system command line
    parameters userParam working directory /home/guest001/
    on attention QUIESCED_SHARE on aix1 user guest001 using passw0rd')
db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status
    add action script /home/guest001/script1 type operating system command line
    parameters userParam working directory /home/guest001/
    on attention QUIESCED_UPDATE on aix1 user guest001 using passw0rd')
```

Health indicators

The health monitor uses health indicators to evaluate the health of specific aspects of database manager performance or database performance. A health indicator measures the health of some aspect of a particular class of database objects, such as table spaces. Criteria are applied to the measurement to determine healthiness. The criteria applied depends on the type of health indicator. A determination of unhealthiness is based on the criteria generates an alert.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Three types of health indicators are returned by the health monitor:

- **Threshold-based** indicators are measurements that represent a statistic (on a continuous range of values) of the behavior of the object. Warning and alarm threshold values define boundaries or zones for normal, warning, and alarm ranges. Threshold-based health indicators have three valid states: Normal, Warning, or Alarm.
- **State-based** indicators are measurements that represent a finite set of two or more distinct states of an object that defines whether the database object or resource is operating normally. One of the states is normal and all others are considered non-normal. State-based health indicators have two valid states: Normal, Attention.
- **Collection state-based** indicators are database-level measurements that represent an aggregate state or one or more objects within the database. Data is captured for each object in the collection and the highest severity of conditions among those objects is represented in the aggregated state. If one or more objects in the collection are in a state requiring an alert, the health indicator shows Attention state. Collection state-based health indicators have two valid states: Normal, Attention.

Health indicators exist at the instance, database, table space, and table space container level.

You can access health monitor information through the Health Center, the CLP, or APIs. You can configure health indicators through these same tools.

An alert is generated in response to either a change from a normal to a non-normal state or a change in the health indicator value to a warning or alarm zone that is based on defined threshold boundaries. There are three types of alerts: attention, warning, and alarm.

- For health indicators measuring distinct states, an attention alert is issued if a non-normal state is registered.
- For health indicators measuring a continuous range of values, threshold values define boundaries or zones for normal, warning and alarm states. For example,

if the value enters the threshold range of values that defines an alarm zone, an alarm alert is issued to indicate that the problem needs immediate attention.

The health monitor will only send notification and run an action on the first occurrence of a particular alert condition for a given health indicator. If the health indicator stays in a particular alert condition, no further notification will be sent and no further actions will be run. If the health indicator changes alert conditions, or goes back to normal state and re-enters the alert condition, notification will be sent anew and actions will be run.

The following table shows an example of a health indicator at different refresh intervals and the health monitor response to the health indicator state. This example uses the default warning of 80% and alarm thresholds of 90%.

Table 36. Health indicator conditions at different refresh intervals

Refresh interval	Value of ts.ts_util (Table space utilization) health indicator	State of ts.ts_util health indicator	Health monitor response
1	80	warning	notification of warning is sent, actions for a warning alert condition are run
2	81	warning	no notification is sent, no actions are run
3	75	normal	no notification is sent, no actions are run
4	85	warning	notification of warning is sent, actions for a warning alert condition are run
5	90	alarm	notification of alarm is sent, actions for an alarm condition are run

Health monitor interface mappings to logical data groups

The following table lists all the supported health snapshot request types.

Table 37. Health monitor interface mappings to logical data groups

API request type	CLP command	SQL table function	Logical data groups
SQLMA_DB2	get health snapshot for dbm	HEALTH_DBM_INFO	db2
		HEALTH_DBM_HI	health_indicator
	get health snapshot for dbm show detail	HEALTH_DBM_HI_HIS	health_indicator_history
SQLMA_DBASE	get health snapshot for database on *dbname*	HEALTH_DB_INFO	dbase
		HEALTH_DB_HI	health_indicator
	get health snapshot for database on *dbname* show detail	HEALTH_DB_HI_HIS	health_indicator_history

Table 37. Health monitor interface mappings to logical data groups (continued)

API request type	CLP command	SQL table function	Logical data groups
SQLMA_DBASE with SQLM_HMON_OPT_COLL_FULL in the agent_id	get health snapshot for database on *dbname* with full collection	HEALTH_DB_HIC	health_indicator, hi_obj_list
	get health snapshot for database on *dbname* show detail with full collection	HEALTH_DB_HIC_HIST	health_indicator_history, hi_obj_list
SQLMA_DBASE_ALL	get health snapshot for all databases	HEALTH_DB_INFO	dbase
		HEALTH_DB_HI	health_indicator
	get health snapshot for all databases show detail	HEALTH_DB_HI_HIS	health_indicator_history
SQLMA_DBASE_TABLESPACES	get health snapshot for tablespaces on *dbname*	HEALTH_TS_INFO	tablespace
		HEALTH_TS_HI	health_indicator
		HEALTH_CONT_INFO	tablespace_container
		HEALTH_CONT_HI	health_indicator
	get health snapshot for tablespaces on *dbname* show detail	HEALTH_TS_HI_HIS	health_indicator_history
		HEALTH_CONT_HI_HIS	health_indicator_history

The following figure shows the order that logical data groupings can appear in a health snapshot data stream.

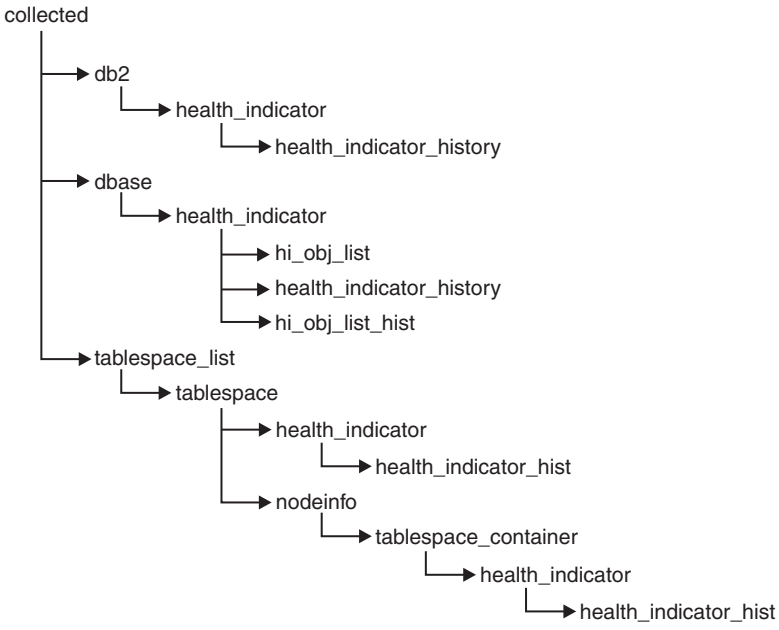


Figure 17. Health snapshot logical data groupings

Health indicators summary

The following tables list all health indicators, grouped by category.

Table 38. Database automatic storage utilization health indicators

Name	Identifier	Additional Information
Database Automatic Storage Utilization	db.auto_storage_util	"db.auto_storage_util - Database automatic storage utilization health indicator" on page 339

Table 39. Table space storage health indicators

Name	Identifier	Additional Information
Table Space Automatic Resize Status	ts.ts_auto_resize_status	"ts.ts_auto_resize_status - Table space automatic resize status health indicator" on page 340
Automatic Resize Table Space Utilization	ts.ts_util_auto_resize	"ts.ts_util_auto_resize - Automatic resize table space utilization health indicator" on page 341
Table Space Utilization	ts.ts_util	"ts.ts_util - Table Space Utilization" on page 341
Table Space Container Utilization	tsc.tscont_util	"tsc.tscont_util - Table Space Container Utilization" on page 342
Table Space Operational State	ts.ts_op_status	"ts.ts_op_status - Table Space Operational State" on page 343
Table Space Container Operational State	tsc.tscont_op_status	"tsc.tscont_op_status - Table Space Container Operational State" on page 343
Table Space Automatic Resize Status	ts.ts_auto_resize_status	"ts.ts_auto_resize_status - Table space automatic resize status health indicator" on page 340

Table 40. Sorting health indicators

Name	Identifier	Additional Information
Private Sort Memory Utilization	db2.sort_privmem_util	"db2.sort_privmem_util - Private Sort Memory Utilization" on page 344
Shared Sort Memory Utilization	db.sort_shrmem_util	"db.sort_shrmem_util - Shared Sort Memory Utilization" on page 345
Percentage of Sorts That Overflowed	db.spilled_sorts	"db.spilled_sorts - Percentage of Sorts That Overflowed" on page 345
Long Term Shared Sort Memory Utilization	db.max_sort_shrmem_util	"db.max_sort_shrmem_util - Long Term Shared Sort Memory Utilization" on page 346

Table 41. Database manager health indicators

Name	Identifier	Additional Information
Instance Operational State	db2.db2_op_status	"db2.db2_op_status - Instance Operational State" on page 347
Instance Highest Severity Alert State	–	"Instance Highest Severity Alert State" on page 347

Table 42. Database health indicators

Name	Identifier	Additional Information
Database Operational State	db.db_op_status	"db.db_op_status - Database Operational State" on page 348
Database Highest Severity Alert State	–	"Database Highest Severity Alert State" on page 348

Table 43. Maintenance health indicators

Name	Identifier	Additional Information
Reorganization Required	db.tb_reorg_req	"db.tb_reorg_req - Reorganization Required" on page 349
Statistics Collection Required health indicator	db.tb_runstats_req	"db.tb_runstats_req - Statistics Collection Required" on page 350
Database Backup Required	db.db_backup_req	"db.db_backup_req - Database Backup Required" on page 350

Table 44. High availability disaster recovery health indicators

Name	Identifier	Additional Information
HADR Operational Status health indicator	db.hadr_op_status	"db.hadr_op_status - HADR Operational Status" on page 351
HADR Log Delay health indicator	db.hadr_delay	"db.hadr_delay - HADR Log Delay" on page 351

Table 45. Logging health indicators

Name	Identifier	Additional Information
Log Utilization	db.log_util	"db.log_util - Log Utilization" on page 352
Log File System Utilization	db.log_fs_util	"db.log_fs_util - Log File System Utilization" on page 352

Table 46. Application concurrency health indicators

Name	Identifier	Additional Information
Deadlock Rate	db.deadlock_rate	"db.deadlock_rate - Deadlock Rate" on page 353
Lock List Utilization	db.locklist_util	"db.locklist_util - Lock List Utilization" on page 354
Lock Escalation Rate	db.lock_escal_rate	"db.lock_escal_rate - Lock Escalation Rate" on page 354
Percentage of Applications Waiting on Locks	db.apps_waiting_locks	"db.apps_waiting_locks - Percentage of Applications Waiting on Locks" on page 355

Table 47. Package cache, catalog cache, and workspace health indicators

Name	Identifier	Additional Information
Catalog Cache Hit Ratio	db.catcache_hitratio	"db.catcache_hitratio - Catalog Cache Hit Ratio" on page 356
Package Cache Hit Ratio	db.pkgcache_hitratio	"db.pkgcache_hitratio - Package Cache Hit Ratio" on page 356

Table 47. Package cache, catalog cache, and workspace health indicators (continued)

Name	Identifier	Additional Information
Shared Workspace Hit Ratio	db.shrworkspace_hitratio	"db.shrworkspace_hitratio - Shared Workspace Hit Ratio" on page 357

Table 48. Memory health indicators

Name	Identifier	Additional Information
Monitor Heap Utilization	db2.mon_heap_util	"db2.mon_heap_util - Monitor Heap Utilization" on page 357
Database Heap Utilization	db.db_heap_util	"db.db_heap_util - Database Heap Utilization" on page 358

Table 49. Federated health indicators

Name	Identifier	Additional Information
Nickname Status	db.fed_nicknames_op_status	"db.fed_nicknames_op_status - Nickname Status" on page 358
Data Source Server Status	db.fed_servers_op_status	"db.fed_servers_op_status - Data Source Server Status" on page 359

Health indicator format:

A description of the data collected by the health indicator.

The documentation for health indicators is described in a standard format as follows:

Identifier
The name of the health indicator. This identifier is used for configuration from the CLP.

Health monitor level
The level at which the health indicator is captured by the health monitor.

Category
The category for the health indicator.

Type The type of the health indicator. There are four possible values for type:

- Upper-bounded threshold-based, where the progression to an alert is: Normal, Warning, Alarm
- Lower-bounded threshold-based
- State-based, where one state is normal and all others are non-normal
- Collection state-based, where the state is based on the aggregation of states from objects in the collection

Unit The unit of the data measured in the health indicator, such as percentage. This is not applicable for state-based or collection state-based health indicators.

Table space storage health indicators:

Health indicators for DMS table spaces:
This table describes which table space health indicators are relevant for a DMS table space based on the characteristics of the table space:

Table 50. Relevant table space health indicators for a DMS table space

Table space characteristics	Maximum table space size defined	Maximum table space size undefined
Automatic resize enabled = Yes	ts.ts_util_auto_resize - Tracks percentage of table space used relative to the maximum defined by you. An alert indicates that the table space will soon be full and requires intervention by you. As long as the maximum size has been set to a reasonable value (that is, the amount of space specified by the maximum size does exist), this is the most important health indicator for this configuration. ts.ts_util - Tracks usage of currently allocated table space storage. An alert may not require intervention by you to resolve any problems since the table space will attempt to increase in size when it is full. ts.ts_auto_resize_status - Tracks health of resize attempts. An alert indicates that the table space failed to resize (that is, the table space is full).	ts.ts_util_auto_resize - Not applicable. No upper bound specified for the table space size. ts.ts_util - Tracks usage of currently allocated table space storage. An alert may not require intervention by you to resolve any problems since the table space will attempt to increase in size. ts.ts_auto_resize_status - Tracks health of resize attempts. An alert indicates that the table space failed to resize (that is, the table space is full). **Note:** If a DMS table space is defined using automatic storage and there is no maximum size specified, you should also pay attention to the db.auto_storage_util health indicator. This health indicator tracks utilization of the space associated with the database storage paths. When this space fills up, the table space is unable to grow. This may result in a table space full condition.
Automatic resize enabled = No	Not a valid configuration. Maximum table space size is only valid for table spaces that have automatic resize enabled.	ts.ts_util_auto_resize - Not applicable. Table space will not attempt to resize. ts.ts_util - Tracks usage of currently allocated table space storage. An alert indicates a table space full condition and requires immediate intervention by you. The table space will not attempt to resize itself. ts.ts_auto_resize_status - Not applicable. Table space will not attempt to resize.

db.auto_storage_util - Database automatic storage utilization health indicator:

This health indicator tracks the consumption of storage for the defined database storage paths.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db.auto_storage_util

Health monitor level
Database

Category
Database

Type Upper-bounded threshold-based

Unit Percentage

When automatic storage table spaces are created, containers are allocated automatically for these table spaces on the database storage paths. If there is no more space on any of the file systems on which the database storage paths are defined, automatic storage table spaces will be unable to increase in size and may become full.

The indicator is calculated using the formula:

```
(db.auto_storage_used / db.auto_storage_total) * 100
```

where

- *db.auto_storage_used* is the sum of used space across all physical file systems identified in the list of database storage paths
- *db.auto_storage_total* is the sum of total space across all physical file systems identified in the list of database storage paths

Database automatic storage path utilization is measured as a percentage of the space consumed on the database storage path file systems, where a high percentage indicates less than optimal function for this indicator.

The "Time to fullness" in the Additional Information line returned for this health indicator is a prediction of how much time is remaining until the maximum size for the table space has been reached.

ts.ts_auto_resize_status - Table space automatic resize status health indicator:

This health indicator identifies whether table space resize operations are succeeding for DMS table spaces which have automatic resize enabled. When a DMS table space with automatic resize enabled fails to increase in size, it is effectively full. This condition may be due to lack of free space on the file systems on which the table space containers are defined, or a result of the table space automatic resize settings. For example, the defined maximum size may have been reached, or the increase amount may be set too high to be accommodated by the remaining free space.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
ts.ts_auto_resize_status

Health monitor level
Table Space

Category
Table Space Storage

Type State-based

Unit Not applicable

ts.ts_util_auto_resize - Automatic resize table space utilization health indicator:

This health indicator tracks the consumption of storage for each automatic resize DMS table space on which a maximum size has been defined. The DMS table space is considered full when the maximum size has been reached.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
ts.ts_util_auto_resize

Health monitor level
Table Space

Category
Table Space Storage

Type Upper-bounded threshold-based

Unit Percentage

The indicator is calculated using the formula:

```
((ts.used * ts.page_size) / ts.max_size) * 100
```

where

- *ts.used* is the value of "tablespace_used_pages - Used pages in table space monitor element" on page 974
- *ts.page_size* is the value of "tablespace_page_size - Table space page size monitor element" on page 963
- *ts.max_size* is the value of "tablespace_max_size - Maximum table space size" on page 960

Automatic resize DMS table space utilization is measured as a percentage of the maximum table space storage consumed. A high percentage indicates the table space is approaching fullness. The short term and long term growth rates, included in the additional information for this indicator, can be used to determine if the current rate of growth is a short term aberration, or consistent with long term growth.

The "Time to fullness" in the Additional Information line returned for this health indicator is a prediction of how much time is remaining until the maximum size for the table space has been reached.

ts.ts_util - Table Space Utilization:

This health indicator tracks the consumption of storage for each DMS table space.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
ts.ts_util

Health monitor level
Table Space

Category
Table Space Storage

Type Upper-bounded threshold-based

Unit Percentage

The DMS table space is considered full when all containers are full.

If automatic resize is enabled on the table space, this health indicator will not be evaluated. Instead, the database automatic storage utilization **db.auto_storage_util** and table space automatic resize status (**ts.ts_auto_resize_status**) health indicators are relevant for table space storage monitoring. The automatic resize table space utilization (**ts.ts_util_auto_resize**) health indicator will also be available if a maximum size was defined on this table space. The table space utilization percentage can still be retrieved from column TBSP_UTILIZATION_PERCENT of the TBSP_UTILIZATION administrative view if it is required.

The indicator is calculated using the formula:

```
(ts.used / ts.usable) * 100
```

where

- *ts.used* is the value of "tablespace_used_pages - Used pages in table space monitor element" on page 974
- *ts.usable* is the value of "tablespace_usable_pages - Usable pages in table space monitor element" on page 973

Table space utilization is measured as the percentage of space consumed, where a high percentage indicates less than optimal function for this indicator.

The short term and long term growth rates, included in the additional information for this indicator, can be used to determine if current rate of growth is a short term aberration or consistent with longer term growth.

The "Time to fullness" in the Additional Information line returned for this health indicator is a prediction of how much time is remaining until the maximum size for the table space has been reached.

tsc.tscont_util - Table Space Container Utilization:

This health indicator tracks the consumption of storage for each SMS table space that is not using automatic storage.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
tsc.tscont_util

Health monitor level
Table Space Container

Category
Table Space Storage

Type Upper-bounded threshold-based

Unit Percentage

An SMS table space is considered full if there is no more space on any of the file systems for which containers are defined.

If free space is not available on the file system to expand an SMS container, the associated table space becomes full.

An alert may be issued for each container defined on the file system that is running out of free space.

The indicator is calculated using the formula:
`(fs.used / fs.total)*100`

where `fs` is the file system in which the container resides.

SMS table space utilization is measured as the percentage of space consumed, where a high percentage indicates less than optimal function for this indicator.

The short term and long term growth rates, included in the additional information for this indicator, can be used to determine if current rate of growth is a short term aberration or consistent with longer term growth.

The "Time to fullness" in the Additional Information line returned for this health indicator is a prediction of how much time is remaining until the maximum size for the table space has been reached.

ts.ts_op_status - Table Space Operational State:

The state of a table space can restrict activity or tasks that can be performed. A change from normal to another state may generate an Attention alert.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
ts.ts_op_status

Health monitor level
Table Space

Category
Table Space Storage

Type State-based

Unit Not applicable

tsc.tscont_op_status - Table Space Container Operational State:

This health indicator tracks the accessibility of the table space container. The accessibility of the container can restrict activity or tasks that can be performed. If the container is not accessible, an Attention alert may be generated.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
tsc.tscont_op_status

Health monitor level
Table Space Container

Category
Table Space Storage

Type State-based

Unit Not applicable

Sorting health indicators:

db2.sort_privmem_util - Private Sort Memory Utilization:

This indicator tracks the utilization of the private sort memory. If db2.sort_heap_allocated (system monitor element) >= *sheapthres* (DBM configuration parameter), sorts may not be getting full sort heap as defined by the *sortheap* parameter and an alert may be generated.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db2.sort_privmem_util

Health monitor level
Database

Category
Sorting

Type Upper-bounded threshold-based

Unit Percentage

Sorting is considered healthy if there is sufficient heap space in which to perform sorting and sorts do not overflow unnecessarily.

The indicator is calculated using the formula:
`(db2.sort_heap_allocated / sheapthres)*100`

The Post Threshold Sorts snapshot monitor element measures the number of sorts that have requested heaps after the sort heap threshold has been exceeded. The value of this indicator, shown in the Additional Details, indicates the degree of severity of the problem for this health indicator.

The Maximum Private Sort Memory Used snapshot monitor element maintains a private sort memory high watermark for the instance. The value of this indicator, shown in the Additional Information, indicates the maximum amount of private sort memory that has been in use at any one point in time since the instance was last recycled. This value can be used to help determine an appropriate value for *sheapthres*.

db.sort_shrmem_util - Shared Sort Memory Utilization:

This indicator tracks the utilization of the shared sort memory. The *sheapthres_shr* database configuration parameter is a hard limit. If the allocation is close to the limit, an alert may be generated.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db.sort_shrmem_util

Health monitor level
Database

Category
Sorting

Type Upper-bounded threshold-based

Unit Percentage

Sorting is considered healthy if there is sufficient heap space in which to perform sorting and sorts do not overflow unnecessarily.

The indicator is calculated using the formula:
`(db.sort_shrheap_allocated / sheapthres_shr)*100`

Note that if *sheapthres_shr* is set to 0, then *sheapthres* serves as the shared sortheap threshold.

The Maximum Shared Sort Memory Used snapshot monitor element maintains a shared sort memory high watermark for the database. The value of this indicator, shown in the Additional Information, indicates the maximum amount of shared sort memory that has been in use at any one point in time since the database has been active. This value can be used to help determine an appropriate value for the shared sort memory threshold.

Consider using the self-tuning memory feature to have sort memory resources automatically allocated as required by the current workload. If you have the self tuning memory feature enabled for the sort memory area, you should configure this health indicator to disable threshold checking.

db.spilled_sorts - Percentage of Sorts That Overflowed:

Sorts that overflow to disk can cause significant performance degradation. If this occurs, an alert may be generated.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db.spilled_sorts

Health monitor level
Database

Category
Sorting

Type Upper-bounded threshold-based

Unit Percentage

Sorting is considered healthy if there is sufficient heap space in which to perform sorting and sorts do not overflow unnecessarily.

The indicator is calculated using the formula:

$$(db.sort_overflows_{t} - db.sort_overflows_{t-1}) / (db.total_sorts_{t} - db.total_sorts_{t-1}) * 100$$

where *t* is the current snapshot and *t-1* is a snapshot 1 hour ago. The system monitor element db.sort_overflows (based on the sort_overflows monitor element) is the total number of sorts that ran out of sort heap and may have required disk space for temporary storage. The element db.total_sorts (based on the total_sorts monitor element) is the total number of sorts that have been executed.

Consider using the self-tuning memory feature to have sort memory resources automatically allocated as required by the current workload. If you have the self tuning memory feature enabled for the sort memory area, you should configure this health indicator to disable threshold checking.

db.max_sort_shrmem_util - Long Term Shared Sort Memory Utilization:

This indicator tracks an over-configured shared sort heap, looking to see if there are resources that can be freed for use somewhere else in the DB2 database system.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db.max_sort_shrmem_util

Health monitor level
Database

Category
Sorting

Type Lower-bounded threshold-based

Unit Percentage

Sorting is considered healthy if there is sufficient heap space in which to perform sorting and if sorts do not overflow unnecessarily.

An alert might be generated when the percentage usage is low.

The indicator is calculated using the formula:
`(db.max_shr_sort_mem / sheapthres_shr)*100`

The system monitor element db.max_shr_sort_mem (based on the sort_shrheap_top monitor element) is the high watermark for shared sort memory usage.

Consider using the self-tuning memory feature to have sort memory resources automatically allocated as required by the current workload. If you have the self-tuning memory feature enabled for the sort memory area, you should configure this health indicator to disable threshold checking.

Database manager (DBMS) health indicators:

db2.db2_op_status - Instance Operational State:

An instance is considered healthy if the instance state does not restrict activity or tasks being performed.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db2.db2_op_status

Health monitor level
Instance

Category
DBMS

Type State-based

Unit Not applicable

The state can be one of the following: Active, Quiesce pending, Quiesced, or Down. A non-Active state may generate an Attention alert.

The health monitor is unable to execute actions for the db2.db2_op_status health indicator if the indicator enters the down state. This state can arise, for example, when an instance that the indicator is monitoring becomes inactive because of an explicit stop request or an abnormal termination. If you want to have the instance restart automatically after any abnormal termination, you can configure the fault monitor (**db2fm**) to keep the instance highly available.

Instance Highest Severity Alert State:

This indicator represents the rolled-up alert state of an instance being monitored. The alert state of an instance is the highest alert state of the instance and its databases, and database objects being monitored.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
Not applicable. This health indicator does not have configuration or recommendations support.

Health monitor level
Instance

Category
DBMS

Type State-based

Unit Not applicable

The order of the alert states is as follows:

- Alarm
- Warning
- Attention
- Normal

The alert state of the instance determines the overall health of the DB2 database system.

Database health indicators:

db.db_op_status - Database Operational State:

The state of the database can restrict activity or tasks that can be performed. The state can be one of the following: Active, Quiesce pending, Quiesced, or Rollforward. A change from Active to another state may generate an Attention alert.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db.db_op_status

Health monitor level
Database

Category
Database

Type State-based

Unit Not applicable

Database Highest Severity Alert State:

This indicator represents the rolled-up alert state of the database being monitored. The alert state of a database is the highest alert state of the database and its objects.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
Not applicable. This health indicator does not have configuration or recommendations support.

Health monitor level
Database

Category
Database

Type State-based

Unit Not applicable

The order of the alert states is as follows:

- Alarm
- Warning
- Attention
- Normal

Maintenance health indicators:

db.tb_reorg_req - Reorganization Required:

This health indicator tracks the need to reorganize tables or indexes within a database. Tables or all indexes defined on a table require reorganization to eliminate fragmented data. The reorganization is accomplished by compacting the information and reconstructing the rows or index data. The result might yield an improved performance and freed space in the table or indexes.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db.tb_reorg_req

Health monitor level
Database

Category
Database Maintenance

Type Collection state-based

Unit Not applicable

You can filter the set of tables evaluated by this health indicator by specifying in your automatic maintenance policy the names of the tables to be evaluated. This can be done using the Automatic Maintenance wizard.

An attention alert might be generated to indicate that reorganization is required. Reorganization can be automated by setting the AUTO_REORG database configuration parameter to ON. If automatic reorganization is enabled, the attention alert indicates either that one or more automatic reorganizations could not complete successfully or that there are tables which require reorganization, but automatic reorganization is not being performed because the size of the table per database partition exceeds the maximum size criteria for tables that should be

considered for offline reorganization. Refer to the collection details of this health indicator for the list of objects that need attention.

db.tb_runstats_req - Statistics Collection Required:

This health indicator tracks the need to collect statistics for tables and their indexes within a database. Tables and all indexes defined on a table require statistics to improve query execution time.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db.tb_runstats_req

Health monitor level
Database

Category
Database Maintenance

Type Collection state-based

Unit Not applicable

The tables considered by this health indicator can be limited using an SQL query. The scope in the additional information displays the subselect clause on system tables for this query.

An attention alert may be generated to indicate that statistics collection is required. Statistics can be automatically collected by setting the AUTO_RUNSTATS database configuration parameter to ON. If automatic statistics collection is enabled, the attention alert indicates that one or more automatic statistics collection actions did not complete successfully.

db.db_backup_req - Database Backup Required:

This health indicator tracks the need for a backup on the database. Backups should be taken regularly as part of a recovery strategy to protect your data against the possibility of loss in the event of a hardware or software failure.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db.db_backup_req

Health monitor level
Database

Category
Database Maintenance

Type State-based

Unit Not applicable

This health indicator determines when a database backup is required based on the time elapsed and amount of data changed since the last backup.

An attention alert might be generated to indicate that a database backup is required. Database backups can be automated by setting the AUTO_DB_BACKUP database configuration parameter to ON. If automatic database backups are enabled, the attention alert indicates that one or more automatic database backups did not complete successfully.

High availability disaster recovery (HADR) health indicators:

db.hadr_op_status - HADR Operational Status:

This health indicator tracks the high availability disaster recovery (HADR) operational state of the database. The state between primary and standby servers can be one of the following: Connected, Congested or Disconnected. A change from Connected to another state might generate an Attention alert.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db.hadr_op_status

Health monitor level
Database

Category
High availability disaster recovery

Type State-based

Unit Not applicable

db.hadr_delay - HADR Log Delay:

This health indicator tracks the current average delay (in minutes) between the data changes on the primary database and the replication of those changes on the standby database. With a large delay value, data loss can occur when failing over to the standby database after a failure on the primary database. A large delay value can also mean longer downtime when takeover is required, because the primary database is ahead of the standby database.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db.hadr_delay

Health monitor level
Database

Category
High availability disaster recovery

Type Upper-bounded threshold-based

Unit Minutes

Logging health indicators:

db.log_util - Log Utilization:

This indicator tracks the total amount of active log space used in bytes in the database.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db.log_util

Health monitor level
Database

Category
Logging

Type Upper-bounded threshold-based

Unit Percentage

Log utilization is measured as the percentage of space consumed, where a high percentage may generate an alert.

The indicator is calculated using the formula:

```
(db.total_log_used / (db.total_log_used + db.total_log_available)) * 100
```

The values for the log-related database configuration parameters, shown in the additional information, display the current allocations for logs. The additional information also includes the application id for the application which has the oldest active transaction. This application can be forced to free up log space.

db.log_fs_util - Log File System Utilization:

Log File System Utilization tracks the fullness of the file system on which the transaction logs reside.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db.log_fs_util

Health monitor level
Database

Category
Logging

Type Upper-bounded threshold-based

Unit Percentage

The DB2 database system may not be able to create a new log file if there is no room on the file system.

Log utilization is measured as the percentage of space consumed. If the amount of free space in the file system is minimal (that is, there is a high percentage for utilization), an alert may be generated.

The indicator is calculated using the formula: *(fs.log_fs_used / fs.log_fs_total)*100* where fs is the file system on which the log resides.

The values for the log-related database configuration parameters, shown in the additional information, display the current allocations for logs. The additional details also shows if user exit is enabled.

If Block on Log Disk Full, shown in the additional details, is set to yes and utilization is at 100%, you should resolve any alerts as soon as possible to limit the impact to applications which cannot commit transactions until the log file is successfully created.

Application concurrency health indicators:

db.deadlock_rate - Deadlock Rate:

Deadlock rate tracks the rate at which deadlocks are occurring in the database and the degree to which applications are experiencing contention problems.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db.deadlock_rate

Health monitor level
Database

Category
Application Concurrency

Type Upper-bounded threshold-based

Unit Deadlocks per hour

Deadlocks may be caused by the following situations:

- Lock escalations are occurring for the database
- An application may be locking tables explicitly when system-generated row locks may be sufficient
- An application may be using an inappropriate isolation level when binding
- Catalog tables are locked for repeatable read
- Applications are getting the same locks in different orders, resulting in deadlock.

The indicator is calculated using the formula:

$(db.deadlocks_{t} - db.deadlocks_{t-1})$

where *t* is the current snapshot and *t-1* is the last snapshot, taken 60 minutes before the current snapshot.

The higher the rate of deadlocks, the greater the degree of contention which may generate an alert.

db.locklist_util - Lock List Utilization:

This indicator tracks the amount of lock list memory that is being used.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db.locklist_util

Health monitor level
Database

Category
Application Concurrency

Type Upper-bounded threshold-based

Unit Percentage

There is one lock list per database and it contains the locks held by all applications concurrently connected to the database. There is a set limit on lock list memory. Once the limit is reached, performance degrades because of the following situations:

- Lock escalation converts row locks to table locks, thereby reducing concurrency on shared objects in the database.
- More deadlocks between applications can occur since applications are waiting for a limited number of table locks. As a result, transactions are rolled back.

An error is returned to the application when the maximum number of lock requests has reached the limit set for the database.

The indicator is calculated using the formula:

```
(db.lock_list_in_use / (locklist * 4096)) * 100
```

Utilization is measured as a percentage of memory consumed, where a high percentage represents an unhealthy condition.

Consider using the self-tuning memory feature to have lock memory resources automatically allocated as required by the current workload. If you have the self-tuning memory feature enabled for the lock memory area, you should configure this health indicator to disable threshold checking.

db.lock_escal_rate - Lock Escalation Rate:

This indicator tracks the rate at which locks have been escalated from row locks to a table lock thereby impacting transaction concurrency.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db.lock_escal_rate

Health monitor level
Database

Category
Application Concurrency

Type Upper-bounded threshold-based

Unit Lock escalations per hour

A lock is escalated when the total number of locks held by an application reaches the maximum amount of lock list space available to the application, or the lock list space consumed by all applications is approaching the total lock list space. The amount of lock list space available is determined by the *maxlocks* and *locklist* database configuration parameters.

When an application reaches the maximum number of locks allowed and there are no more locks to escalate, the application uses the space in the lock list allocated for other applications. There is one lock list per database and it contains the locks held by all applications concurrently connected to the database. When the entire lock list is full, an error occurs.

The indicator is calculated using the formula:
$(db.lock_escals_{t} - db.lock_escals_{t-1})$

where 't' is the current snapshot and 't-1' is the last snapshot, taken 60 minutes before the current snapshot.

The higher the rate of deadlocks, the greater the degree of contention which may generate an alert.

Consider using the self-tuning memory feature to have lock memory resources automatically allocated as required by the current workload. If you have the self-tuning memory feature enabled for the lock memory area, you should configure this health indicator to disable threshold checking.

db.apps_waiting_locks - Percentage of Applications Waiting on Locks:

This indicator measures the percentage of all currently executing applications that are waiting on locks.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db.apps_waiting_locks

Health monitor level
Database

Category
Application Concurrency

Type Upper-bounded threshold-based

Unit Percentage

A high percentage can indicate that applications are experiencing concurrency problems which can negatively affect performance.

The indicator is calculated using the formula:
`(db.locks_waiting / db.appls_cur_cons) *100)`

Package cache, catalog cache, and workspace health indicators:

db.catcache_hitratio - Catalog Cache Hit Ratio:

The hit ratio is a percentage indicating how well the catalog cache is helping to avoid actual accesses to the catalog on disk. A high ratio indicates it is successful in avoiding actual disk I/O accesses.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db.catcache_hitratio

Health monitor level
Database

Category
Package and Catalog Caches, and Workspaces

Type Lower-bounded threshold-based

Unit Percentage

The indicator is calculated using the formula:
`(1-(db.cat_cache_inserts/db.cat_cache_lookups))*100`

db.pkgcache_hitratio - Package Cache Hit Ratio:

The hit ratio is a percentage indicating how well the package cache is helping to avoid reloading packages and sections for static SQL from the system catalogs as well as helping to avoid recompiling dynamic SQL statements. A high ratio indicates it is successful in avoiding these activities.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db.pkgcache_hitratio

Health monitor level
Database

Category
Package and Catalog Caches, and Workspaces

Type Lower-bounded threshold-based

Unit Percentage

The indicator is calculated using the formula:

```
(1-(db.pkg_cache_inserts/db.pkg_cache_lookups))*100
```

Consider using the self-tuning memory feature to have package cache memory resources automatically allocated as required by the current workload. If you have the self-tuning memory feature enabled for the package cache memory area, you should configure this health indicator to disable threshold checking.

db.shrworkspace_hitratio - Shared Workspace Hit Ratio:

The hit ratio is a percentage indicating how well the shared SQL workspace is helping to avoid having to initialize sections for SQL statements that are about to be executed. A high ratio indicates it is successful in avoiding this action.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Note: The **db.shrworkspace_hitratio** health indicator is deprecated starting with DB2 Version 9.5. Using this health indicator will not generate an error. However, it does not return a valid value. This indicator is no longer recommended and might be removed in a future release.

Identifier
db.shrworkspace_hitratio

Health monitor level
Database

Category
Package and Catalog Caches, and Workspaces

Type Lower-bounded threshold-based

Unit Percentage

The indicator is calculated using the formula:

```
(1-(db.shr_workspace_section_inserts/db.shr_workspace_section_lookups))*100
```

Memory health indicators:

db2.mon_heap_util - Monitor Heap Utilization:

This indicator tracks the consumption of the monitor heap memory, based on the memory pool with the ID SQLM_HEAP_MONITOR.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db2.mon_heap_util

Health monitor level
Instance

Category
Memory

Type Upper-bounded threshold-based

Unit Percentage

The utilization is calculated using the formula:
`(db2.pool_cur_size / db2.pool_config_size) * 100`

for the Memory Pool Identifier SQLM_HEAP_MONITOR.

Once this percentage reaches the maximum, 100%, monitor operations may fail.

db.db_heap_util - Database Heap Utilization:

This indicator tracks the consumption of the monitor heap memory, based on the memory pool with the ID SQLM_HEAP_DATABASE.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db.db_heap_util

Health monitor level
Database

Category
Memory

Type Upper-bounded threshold-based

Unit Percentage

The utilization is calculated using the formula
`(db.pool_cur_size / db.pool_config_size) * 100`

for the Memory Pool Identifier SQLM_HEAP_DATABASE.

Once this percentage reaches the maximum, 100%, queries and operations may fail because there is no heap available.

Federated health indicators:

db.fed_nicknames_op_status - Nickname Status:

This health indicator checks all of the nicknames defined in a federated database to determine if there are any invalid nicknames. A nickname may be invalid if the data source object was dropped or changed, or if the user mapping is incorrect.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db.fed_nicknames_op_status

Health monitor level
Database

Category
Federated

Type Collection state-based

Unit Not applicable

An attention alert might be generated if any nicknames defined in the federated database are invalid. Refer to the collection details of this health indicator for the list of objects that need attention.

The FEDERATED database manager parameter must be set to YES for this health indicator to check nicknames status.

db.fed_servers_op_status - Data Source Server Status:

This health indicator checks all of the data source servers defined in a federated database to determine if any are unavailable. A data source server might be unavailable if the data source server was stopped, no longer exists, or was incorrectly configured.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

Identifier
db.fed_servers_op_status

Health monitor level
Database

Category
Federated

Type Collection state-based

Unit Not applicable

An attention alert might be generated if any nicknames defined in the federated database are not valid. Refer to the collection details of this health indicator for the list of objects that need attention.

The FEDERATED database manager parameter must be set to YES for this health indicator to check data source server status.

Health monitor interfaces

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

The following table lists the health monitor interfaces for APIs:

Table 51. Health monitor interfaces: APIs

Monitoring task	API
Capturing a health snapshot	db2GetSnapshot - Get Snapshot with snapshot class SQLM_CLASS_HEALTH

Table 51. Health monitor interfaces: APIs (continued)

Monitoring task	API
Capturing a health snapshot with the full list of collection objects	db2GetSnapshot - Get Snapshot with snapshot class SQLM_CLASS_HEALTH and SQLM_HMON_OPT_COLL_FULL for agent_id
Capturing a health snapshot with formula, additional information, and history	db2GetSnapshot - Get Snapshot with snapshot class SQLM_CLASS_HEALTH_WITH_DETAIL
Capturing a health snapshot with formula, additional information, history, and the full list of collection objects	db2GetSnapshot - Get Snapshot with snapshot class SQLM_CLASS_HEALTH_WITH_DETAIL and SQLM_HMON_OPT_COLL_FULL for agent_id
Converting the self-describing data stream	db2ConvMonStream - Convert Monitor stream
Estimating the size of a health snapshot	db2GetSnapshotSize - Estimate Size Required for db2GetSnapshot Output Buffer

The **GET HEALTH SNAPSHOT** command is part of the health monitor components that have been deprecated.

The following table lists the health monitor interfaces for CLP commands:

Table 52. Health monitor interfaces: CLP commands

Monitoring task	CLP command
Capturing a health snapshot	GET HEALTH SNAPSHOT Command
Capturing a health snapshot with formula, additional information, and history	GET HEALTH SNAPSHOT WITH DETAILS Command

The health monitor SQL functions are part of the health monitor components that have been deprecated.

The following table lists the health monitor interfaces for SQL functions:

Table 53. Health monitor interfaces: SQL functions

Monitoring task	SQL Function
Database manager level health information snapshot	HEALTH_DBM_INFO
Database manager level health indicator snapshot	HEALTH_DBM_HI
Database manager level health indicator history snapshot	HEALTH_DBM_HI_HIS
Database level health information snapshot	HEALTH_DB_INFO
Database level health indicator snapshot	HEALTH_DB_HI
Database level health indicator history snapshot	HEALTH_DB_HI_HIS
Database level health indicator collection snapshot	HEALTH_DB_HIC
Database level health indicator collection history snapshot	HEALTH_DB_HIC_HIS
Table space level health information snapshot	HEALTH_TBS_INFO
Table space level health indicator snapshot	HEALTH_TBS_HI
Table space level health indicator history snapshot	HEALTH_TBS_HI_HIS
Table space container level health information snapshot	HEALTH_CONT_INFO
Table space container level health indicator snapshot	HEALTH_CONT_HI
Table space container level health indicator history snapshot	HEALTH_CONT_HI_HIS

Health monitor SQL table functions:

Health monitor SQL table functions return information from a health snapshot of a DB2 database manager, database, or its table spaces.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

The following table lists all of the snapshot table functions. Each table function corresponds to a health snapshot request type.

Table 54. Snapshot monitor SQL table functions

Monitor level	SQL table function	Information returned
Database manager	HEALTH_DBM_INFO	Basic information about the health snapshot from the database manager level
Database manager	HEALTH_DBM_HI	Health indicator information from the database manager level
Database manager	HEALTH_DBM_HI_HIS	Health indicator history information from the database manager level
Database	HEALTH_DB_INFO	Basic information about the health snapshot from a database
Database	HEALTH_DB_HI	Health indicator information from a database
Database	HEALTH_DB_HI_HIS	Health indicator history information from a database
Database	HEALTH_DB_HIC	Collection information for collection health indicators for a database
Database	HEALTH_DB_HIC_HIS	Collection history information for collection health indicators for a database
Table space	HEALTH_TBS_INFO	Basic information about the health snapshot for the table spaces for a database
Table space	HEALTH_TBS_HI	Health indicator information about the table spaces for a database
Table space	HEALTH_TBS_HI_HIS	Health indicator history information about the table spaces for a database
Table space	HEALTH_CONT_INFO	Basic information about the health snapshot for the containers for a database
Table space	HEALTH_CONT_HI	Health indicator information about the containers for a database
Table space	HEALTH_CONT_HI_HIS	Health indicator history information about the containers for a database

Health monitor CLP commands:

You can get health status information for the database manager and its databases by issuing health monitor commands.

The information returned represents a snapshot of the health state at the time the command was issued.

Important: The health monitor, health indicators, and related components have been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Health monitor has been deprecated" topic in the *What's New for DB2 Version 9.7* book.

The following table lists all the supported snapshot request types.

Table 55. Snapshot monitor CLP commands

Monitor level	CLP command	Information returned
Database manager	`get health snapshot for dbm`	Database manager level information.
Database	`get health snapshot for all databases`	Database level information. Information is returned only if the database is activated.
Database	`get health snapshot for database on database-alias`	Database level information. Information is returned only if the database is activated.
Database	`get health snapshot for all on database-alias`	Database, table space, and table space container information. Information is returned only if the database is activated.
Table space	`get snapshot for tablespaces on database-alias`	Table space level information for each table space that was accessed by an application connected to the database. Also includes health information for each table space container within the table space.

Health monitor API request types:

The following table lists all the supported snapshot request types.

Table 56. Snapshot Monitor API Request Types

Monitor level	API request type	Information returned
Database manager	SQLMA_DB2	Database manager level information.
Database	SQLMA_DBASE_ALL	Database level information. Information is returned only if the database is activated.
Database	SQLMA_DBASE	Database level information. Information is returned only if the database is activated.
Table space	SQLMA_DBASE_TABLESPACES	Table space level information for each table space that has been accessed by an application connected to the database. Also includes health information for each table space container within the table space.

Working with the Memory Visualizer

The Memory Visualizer helps database administrators to monitor the memory-related performance of an instance and all of its databases. You can view a live, visual display of the memory utilization of memory components organized in a hierarchical tree.

Before you begin

Important: The Memory Visualizer has been deprecated in Version 9.7 and might be removed in a future release. Start using IBM Data Studio and IBM Optim tools.

For a mapping between these recommended tools and Control Center tools, see "Table of recommended tools versus Control Center tools" in the *What's New for DB2 Version 9.7* book.

To view memory performance and usage plots and to update the configuration parameters in the Memory Visualizer, you must have SYSADM authority.

About this task

You can use the Memory Visualizer to troubleshoot performance problems. You can change the configuration parameter settings for a memory component and assess the effect that the changes have. Configuration parameters affect memory usage in DB2 because memory is allocated as it is required. If you set the value of a configuration parameter above or below its acceptable range, an error message will display. Changing the configuration parameter takes immediate effect within the Memory Visualizer, and the new value is integrated in the next refresh cycle.

Procedure

- To view memory performance using the Memory Visualizer:
  1. Open the Memory Visualizer from the Windows**Start** menu, by clicking **Programs** → **IBM DB2** → **Monitoring** → **Tools** → **Memory Visualizer**. The Memory Visualizer instance selection window opens. Select an instance from the **Instance name** field and click **OK**.
  2. Expand the instance object tree until you display the databases and their associated memory components in the hierarchical tree. Values for the memory pools display in the Memory Visualizer window.
  3. To display a plotted graph of a memory component, use one of the following methods:
     - Select a component in the hierarchical tree and click the **Show Plot** check box in the Memory Visualizer window.
     - Right-click the selected memory component to display a pop-up menu and select **Show Plot**.
     - Select a component in the hierarchical tree and select the **Show Plot** option from the Selected menu on the tool bar. The plotted data for each memory component appears in the Memory Usage Plot.
     - To view data from another memory component, select it from the hierarchical tree and click the **Show Plot** check box. The plotted data for the component appears in the Memory Usage Plot along with other components.

     The graph displays data collected for memory components over time. Each component is represented by a color and shape which also displays in the **Plot Legend** field in the Memory Visualizer window. The shape is repeated at intervals. A label identifies the component in the graph plot.

     The time that the performance data was captured is displayed below the graph. You can change the time interval for the graph.

     Note: When a new memory component is added to the plot, it does not replace the memory components that were previously added.

     Horizontal and vertical scroll bars offer different views of the plotted data.

     - Use the horizontal scroll bar, located at the bottom of the graph, to view historical data of the memory component over a selected time period. Point to and drag the slider bar along the base of the graph.

– Use the vertical scroll bar, located at the right of the graph, to view the memory utilization of the selected component. Point to and drag the slider to change the view.

When the memory utilization reaches a new high, the maximum value of the vertical scroll bar is updated to reflect the new value. You can set the minimum value of the vertical scroll bar to a value other than `0` to view a different range of pool utilization values.

- You can load data from a Memory Visualizer data file into a new Memory Visualizer window. This data can be used to compare the performance of an instance and all of its databases against historical data. To load data from a Memory Visualizer data file, select **Open** from the Memory Visualizer menu, and then from the Open Dialog select a data file with extension *.mdf.
- Use the **Time Unit** field to change the time interval on the Memory Usage Plot window. The default time interval for the graph data is minutes. You can select intervals of minutes, hours, or days. When selected, a new time interval displays in the horizontal range of the graph and changes the incremental movement of the horizontal scroll bar.
- To remove the plotted graph of a memory component from the Memory Usage Plot, either select a component in the hierarchical tree and clear the **Show Plot** check box in the Memory Visualizer window, or right-click the selected memory component to display a pop-up menu and deselect **Show Plot**. The plotted data for the component is removed from the Memory Usage Plot window. The colored shape, which represented the component, no longer displays in the **Plot Legend** field in the Memory Visualizer window.
- To help you to track and create a history of memory performance, you can save memory performance data, including plotted graphs, while the Memory Visualizer is running. To save memory performance data, select **Save** or **Save As** from the Memory Visualizer menu, and then select a location for the file and a filename with extension `.mdf`.
- To change the configuration parameter settings for a memory component:
  1. Expand the memory pool that you want so that you can see its configuration parameters listed in the hierarchical tree.
  2. Click a component to select it and click the number in the **Parameter Value** column. A text box displays the current value for the component. Type a new number in the text box and press **Enter**. The new value displays next to the original value in the **Parameter Value** column until the configuration parameter is updated possibly in the next refresh cycle. You can also right-click the value in the **Parameter Value** column for the selected component to display the pop-up menu. Click outside of the column to complete the change. The new value for the memory component displays next to the original value in the **Parameter Value** column. If you select to view a graph of memory performance, you will see the new value in the graph plot view. While this change takes place immediately in the Memory Visualizer, there is a delay in updating the change you made to the configuration parameter within DB2. You can reset the value of the configuration parameter using the **Reset to Default** option in the pop-up menu.

Memory Visualizer overview

Use the Memory Visualizer to monitor the memory-related performance of an instance and all of its databases.

Important: The Memory Visualizer has been deprecated in Version 9.7 and might be removed in a future release. Start using IBM Data Studio and IBM Optim tools. For a mapping between these recommended tools and Control Center tools, see "Table of recommended tools versus Control Center tools" in the *What's New for DB2 Version 9.7* book.

Open the Memory Visualizer and select a memory component or multiple components in the hierarchical tree to display values for the amount of memory allocated to the component and the current memory usage in the Memory Visualizer window. The Memory Visualizer window displays two views of data: a tree view and a historical view. A series of columns show percentage threshold values for upper and lower alarms and warnings. The columns also display real time memory utilization.

Note: The Memory Visualizer is available to provide memory performance data for instances that are Version 8.1 and later.

The following list categorizes some of the key tasks that you can do with the Memory Visualizer:

- View or hide data in various columns on the memory utilization of selected components for a DB2 instance and its databases.
- View a graph of memory performance data.
- Change settings for individual memory components by updating configuration parameters.
- Load performance data from a file into a Memory Visualizer window.
- Save the memory performance data.

The Memory Visualizer interface has the following elements that help you monitor the memory-related performance of an instance and all of its databases.

The Memory Visualizer window

The columns in the Memory Visualizer window display values for the performance of memory components. The following information is shown:

Plot Legend

The checked memory components or configuration parameters shown in the Memory Usage Plot. A specific shape that occurs at regular intervals in the plotted graph identifies each component or parameter.

Utilization

The size of the memory that is allocated to, and utilized by, the database object. Includes a graphical bar showing the utilization and configured allocation. The length of the bar is fixed and the filled portion indicates utilization as a percentage.

Parameter Value

The current value of a configuration parameter.

Upper Alarm (%) Threshold

The threshold value that generates an upper alarm. The default value is 98%.

Upper Warning (%) Threshold

The threshold value that generates an upper warning. The default value is 90%.

Lower Alarm (%) Threshold
The threshold value that generates a lower alarm. The default value is 2%.

Lower Warning (%) Threshold
The threshold value that generates a lower warning. The default value is 10%.

Graphical Usage Bars

The graphical usage bars in the Memory Visualizer window are visual cues of memory utilization. The bars can assist you in determining how much memory is being used by selected memory components and the potential effect that the usage can have on the system. The Memory Visualizer also displays a percentage value that corresponds to the usage. These two indicators can help you to determine whether you need to change the configuration parameter setting for the component or take another appropriate action.

Memory Components

The Database Manager uses different types of memory on a system, namely Database manager shared memory, Database global memory, Application global memory, Agent /Application shared memory, and Agent private memory. These types of memory are the high level memory components that the Memory Visualizer uses in its expanding hierarchical tree organization.

Underlying each high-level memory component are other components that determine how the memory is allocated and deallocated. For example, when the database manager starts, a database is activated, an application connects to a database, or when an agent is assigned to work for an application, memory is allocated and deallocated. The Memory Visualizer uses these leaf-level memory components to display how memory is allocated and used in a DB2 instance.

Hierarchical Tree Organization

The Memory Visualizer uses a hierarchical tree organization to help you to display and browse the memory components in DB2. The hierarchical tree allows you to expand and view information on individual memory components through columns, graphical displays, and graphs.

The tree view comprises four major types of memory items:

DB2 Instance
The instance that is currently running on the system

Databases
The databases defined on the instance

High-level memory components
Logical groupings for leaf-level memory components. These groups are: Database manager shared memory, Database global memory, Agent private memory, Agent /Application shared memory

Leaf-level memory components
The memory components that display in the Memory Visualizer window such as buffer pools, sort heaps, database heap, and lock list.

Icons in the tree view represent each memory tree item:

- Instance:
- Database:
- High-level memory groupings:
- Leaf-level memory components:

If the memory utilization for a tree items exceeds a threshold value, a colored indicator overlays the icon. The yellow color indicates a warning condition. The red color indicates an alarm condition.

The historical view displays data for memory components selected in the tree view. The data includes values for memory allocated and utilized, plotted graphs, as well as changes made to the configuration parameters while the Memory Visualizer is running. The data is saved for a specific period within the Memory Visualizer. You can save memory performance data to a Memory Visualizer data file for tracking, comparing with other data, or troubleshooting.

The Memory Usage Graph

The memory usage graph displays plotted data for selected memory components in the Memory Usage Plot. Each component in the graph is identified by a specific color, which also displays in the **Plot Legend** column in the Memory Visualizer window. The graph also displays changes made to the configuration parameters settings. The original value of the configuration parameter and the new value setting appear in the graph, in addition to the time that the change was requested. They become part of the history view that you can use in assessing memory performance.

For further details, see "Working with the Memory Visualizer" on page 362.

Activity Monitor overview

Use the Activity Monitor to monitor application performance and concurrency, resource consumption, and SQL statement usage of a database or database partition.

Important: The Activity Monitor has been deprecated in Version 9.7 and might be removed in a future release. Start using IBM Data Studio and IBM Optim tools. For a mapping between these recommended tools and Control Center tools, see "Table of recommended tools versus Control Center tools" in the *What's New for DB2 Version 9.7* book.

The Activity Monitor provides a set of predefined reports based on a specific subset of monitor data. These reports allow you to focus monitoring on application performance, application concurrency, resource consumption, and SQL statement use. The Activity Monitor also provides recommendations for most reports. These recommendations can assist you to diagnose the cause of database performance problems, and to tune queries for optimal utilization of database resources.

Figure 18 on page 368 describes the process for using the Activity monitor to solve a problem.

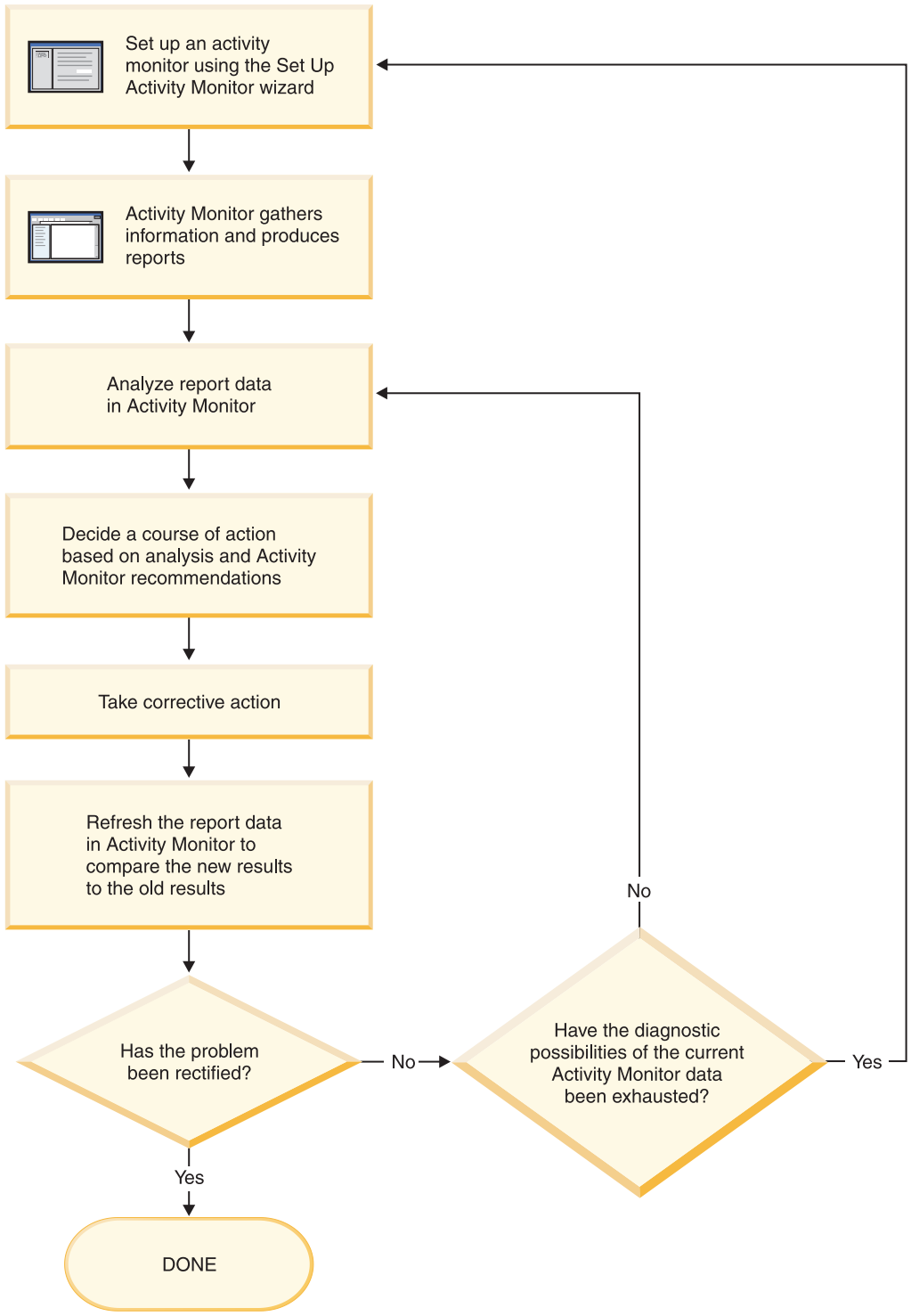


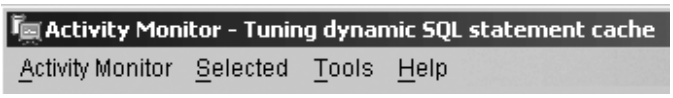
Figure 18. Activity Monitor overview

Table 57. Tasks that you can perform from the Activity Monitor

Tasks from the Activity Monitor	**Aspects of tasks**	**Invocation**
Transactions	View transactions running on a selected application.	Select one or more applications in the **Report data** pane. Right-click and select **Show Latest Transactions** . The Application Transactions window opens.
Statements	View SQL statements running on a selected application.	Select one or more applications in the **Report data** pane. Right-click and select **Show Latest Statements**. The Application Statements window opens.
	View the text of SQL statements running on a selected application.	From the Application Statements window, right-click on a statement in the **Report data** pane. Select **Show Statement Text**
Application Lock Chains	View locks and lock-waiting situations that currently affect a selected application.	Select an application in the **Report data** pane. Right-click and select **Show Lock Chains**. The Application Lock Chains window opens.
	View information about a selected application for which you are viewing lock information.	From the Application Lock Chains window, right-click an application, and select **About**.
	View information about the locks held and the locks waited on by a selected application in your database.	From the Application Lock Chains window, right-click an application, and select **Show Lock Details**.
View report data and recommendations	View information to help you interpret report data.	From an Activity Monitor window, an Application Statements window, or an Application Transactions window, use the **Report** arrow to select the report and click the **Report Details** push button. View the **Details** page.
	View recommendations provided by Activity Monitor	From an Activity Monitor window, an Application Statements window, or an Application Transactions window, use the **Report** arrow to select the report and click the **Report Details** push button. View the **Recommendations** page.

The Activity Monitor interface has several elements that help you organize and interpret the monitor data that is collected:

Menu bar

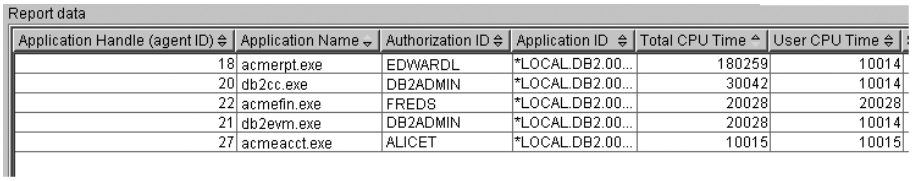


Use the menu bar to work with objects in the Activity Monitor, open other administration centers and tools, and access online help.

Activity Monitor toolbar

Use the toolbar icons to open DB2 tools and view DB2 information.

Report data pane


Report data

Application Handle (agent ID)	Application Name	Authorization ID	Application ID	Total CPU Time	User CPU Time
18	acmerpt.exe	EDWARDL	*LOCAL.DB2.00...	180259	10014
20	db2cc.exe	DB2ADMIN	*LOCAL.DB2.00...	30042	10014
22	acmefin.exe	FREDS	*LOCAL.DB2.00...	20028	20028
21	db2evm.exe	DB2ADMIN	*LOCAL.DB2.00...	20028	10014
27	acmeacct.exe	ALICET	*LOCAL.DB2.00...	10015	10015

Use the **Report data** pane to display and to work with the report data that is available to you within the Activity Monitor. The **Report data** pane displays the items that make up the contents of the report that is selected in the **Report** field.

The **Report data** pane also provides access to other Activity Monitor windows. From the Activity Monitor, you can drill down from the applications you are monitoring to the individual transactions or to the individual SQL statements that these applications are running.

Report data pane toolbar

Use the toolbar located below the **Report data** pane to tailor the view of objects and information in the **Report data** pane to suit your needs.

Monitoring scenarios

Scenario: Identifying costly applications using snapshot administrative views

Recent increases in the workload on the ShopMart database have started hindering overall database performance. Jessie, the ShopMart DBA, is trying to identify the larger resource consumers in the daily workload using the following administrative views:

APPLICATION_PERFORMANCE

This view helps Jessie identify applications that might be performing large table scans:

```
connect to shopmart;
select AGENT_ID, ROWS_SELECTED, ROWS_READ from APPLICATION_PERFORMANCE;
```

The value of ROWS_SELECTED shows her how many rows are returned to an application and the value of ROWS_READ shows her how many rows are accessed from the base tables. If the selectivity is low, the application might be performing a table scan that could be avoided with

the creation of an index. Jessie uses this view to identify potentially troublesome queries, and then she can investigate further by looking at the SQL to see if there are any ways to reduce the number of rows that are read in the execution of the query.

LONG_RUNNING_SQL

Jessie uses the LONG_RUNNING_SQL administrative view to identify the longest running queries that are currently being executed:

```
connect to shopmart;
select ELAPSED_TIME_MIN, APPL_STATUS, AGENT_ID
 from long_running_sql order by ELAPSED_TIME_MIN desc
 fetch first 5 rows only;
```

Using this view, she can determine the length of time these queries have been running, and the status of these queries. If a query has been executing for a long time and is waiting on a lock, she can use the LOCKWAITS or LOCK_HELD administrative views querying on a specific agent id to investigate further. The LONG_RUNNING_SQL view can also tell her the statement that is being executed, allowing her to identify potentially problematic SQL.

QUERY_PREP_COST

Jessie uses the QUERY_PREP_COST to troubleshoot queries that have been identified as problematic. This view can tell her how frequent a query is run as well as the average execution time for each of these queries:

```
connect to shopmart;
select NUM_EXECUTIONS, AVERAGE_EXECUTION_TIME_S, PREP_TIME_PERCENT
 from QUERY_PREP_COST  order by NUM_EXECUTIONS desc;
```

The value of PREP_TIME_PERCENT tells Jessie what percentage of the queries execution time is spent preparing the query. If the time it takes to compile and optimize a query is almost as long as it takes for the query to execute, Jessie might want to advise the owner of the query to change the optimization class used for the query. Lowering the optimization class might make the query complete optimization more rapidly and therefore return a result sooner. However, if a query takes a significant amount of time to prepare but is executed thousands of times (without being prepared again) then changing the optimization class might not benefit query performance.

TOP_DYNAMIC_SQL

Jessie uses the TOP_DYNAMIC_SQL view to identify the most frequently executed, longest-running and most sort-intensive dynamic SQL statements. Having this information will allow Jessie to focus her SQL tuning efforts on the queries that represent some of the biggest resource consumers

To identify the most frequently run dynamic SQL statements, Jessie issues the following:

```
connect to shopmart;
select * from TOP_DYNAMIC_SQL order by NUM_EXECUTIONS desc
fetch first 5 rows only;
```

This returns all of the details regarding the execution time, number of sorts performed, and the statement text for the five most frequent dynamic SQL statements.

To identify the dynamic SQL statements with the longest execution times, Jessie examines the queries with the top five values for AVERAGE_EXECUTION_TIME_S:

```
connect to shopmart;
select * from TOP_DYNAMIC_SQL
 order by AVERAGE_EXECUTION_TIME_S desc fetch first 5 rows only;
```

To look at the details of the most sort-intensive dynamic SQL statements, Jessie issues the following:

```
connect to shopmart;
select STMT_SORTS, SORTS_PER_EXECUTION, substr(STMT_TEXT,1,60) as STMT_TEXT
 from TOP_DYNAMIC_SQL order by STMT_SORTS desc
 fetch first 5 rows only;
```

Scenario: Monitoring buffer pool efficiency using administrative views

John, a DBA, suspects that poor application performance in the SALES database is a result of buffer pools that function inefficiently. To investigate, he takes a look at the buffer pool hit ratio using the BP_HITRATIO administrative view:

```
connect to SALES;
select BPNAME, TOTAL_HIT_RATIO from BP_HIT_RATIO;
```

John sees that the hit ratio for one of the buffer pools is very low, which means that too many pages are being read from disk instead of being read from the buffer pool.

He then decides to use the BP_READ_IO administrative view to see whether the prefetchers require tuning:

```
connect to SALES;
select BPNAME, PERCENT_SYNC_READS,UNUSED_ASYNC_READS_PERCENT from BP_READ_IO;
```

The value for PERCENT_SYNC_READS tells him the percentage of pages read synchronously without prefetching. A high number indicates that a high percentage of data is being read directly from disk, and might indicate that more prefetchers are required. The value of UNUSED_ASYNC_READS_PERCENT tells him the percentage of pages read asynchronously from disk, but never accessed by a query. This might indicate that the prefetchers are overly aggressive in reading in data pages, resulting in unnecessary I/O.

Since both the values for PERCENT_SYNC_READS and UNUSED_ASYNC_READS_PERCENT seem within the acceptable range, John uses the BP_WRITE_IO administrative view to investigate how well the page cleaners are working to clear space for incoming data pages:

```
connect to SALES;
select BPNAME, PERCENT_WRITES_ASYNC from BP_WRITE_IO;
```

The value of PERCENT_WRITES_ASYNC tells John what percentage of physical write requests that were performed asynchronously. If this number is high, it might means that the page cleaners are working well to clear space in the buffer pool ahead of incoming requests for new data pages. If this number is low, then a higher number of physical writes are being performed by database agents while an application waits for data a data page to be read into the buffer pool.

John sees that the value of PERCENT_WRITES_ASYNC is very low at 25 percent, so he decides to configure more page cleaners for the SALES database to increase the rate of asynchronous writes. After increasing the number of page cleaners, he can use the buffer pool administrative views again to see the effects of his tuning.

Setting up an activity monitor

To monitor application performance and concurrency, resource consumption, and SQL statement usage of a database or database partition, you can set up an activity monitor. The Activity Monitor provides a set of predefined reports based on a specific subset of monitor data. It can also provide recommendations to assist you in diagnosing the cause of database performance problems, and to tune queries so that your use of database resources is optimized.

Before you begin

Important: The Activity Monitor has been deprecated in Version 9.7 and might be removed in a future release. Start using IBM Data Studio and IBM Optim tools. For a mapping between these recommended tools and Control Center tools, see "Table of recommended tools versus Control Center tools" in the *What's New for DB2 Version 9.7* book.

To use the Activity Monitor:

- Your server must have DB2 UDB Version 8.2 or later
- You must have SQLADM or DBADM authority

Procedure

Open the Set Up Activity Monitor wizard.

- From the Control Center, expand the object tree until you find the instance or the database for which you want to set up an activity monitor. Right-click the object and select Set Up Activity Monitor from the pop-up menu.
- From the command line, type the following command: **db2am**.

Results

Detailed information is provided through the contextual help facility within the Control Center.

Progress monitoring of the rollback process

If you obtain an application snapshot while a transaction is rolling back, you will see rollback monitor elements in the output. This information can be used to monitor the progress of the rollback operation.

The information provided in the application snapshot includes the start time of the rollback, the total work to be done, and completed work. The work metric is bytes.

The following is an example of output from the **GET SNAPSHOT FOR ALL APPLICATIONS** command:

```
            Application Snapshot

Application handle         = 6
Application status         = Rollback Active
      Start Time           = 02/20/2004 12:49:27.713720
      Completed Work       = 1024000 bytes
      Total Work           = 4084000 bytes

            Application Snapshot

Application handle         = 10
Application status         = Rollback to Savepoint
```

```
Start Time           = 02/20/2004 12:49:32.832410
Completed Work       = 102400 bytes
Total Work           = 2048000 bytes
```

The value in the application status monitor element implies which type of rollback event is occurring:

Rollback Active
This is a unit of work rollback: an explicit (user invoked) or implicit (forced) rollback of the entire transaction.

Savepoint rollback
This is a partial rollback to a statement or application level savepoint. Nested savepoints are considered a single unit, using the outermost savepoint.

Completed Work units shows the relative position in the log stream that has been rolled back. Updates to Completed Work are made after every log record is processed. Updates are not performed evenly because log records vary in size.

Total Work units is an estimate based on the range of log records in the log stream that need to be rolled back for the transaction or savepoint. It does not indicate the exact number of log record bytes that need to be processed.

Using snapshot monitor data to monitor the reorganization of a partitioned table

The following information describes some of the most useful methods of monitoring the global status of a table reorganization.

About this task

There is no separate data group indicating the overall table reorganization status for a partitioned table. A partitioned table uses a data organization scheme in which table data is divided across multiple storage objects, called data partitions or ranges, according to values in one or more table partitioning key columns of the table. However, you can deduce the global status of a table reorganization from the values of elements in the individual data partition data group being reorganized. The following information describes some of the most useful methods of monitoring the global status of a table reorganization.

Determining the number of data partitions being reorganized

You can determine the total number of data partitions being reorganized on a table by counting the number of monitor data blocks for table data that have the same table name and schema name. This value indicates the number of data partitions on which reorganization has started. Examples 1 and 2 indicate that three data partitions are being reorganized.

Identifying the data partition being reorganized

You can deduce the current data partition being reorganized from the phase start time (reorg_phase_start). During the SORT/BUILD/REPLACE phase, the monitor data corresponding to the data partition that is being reorganized shows the most recent phase start time. During the INDEX_RECREATE phase, the phase start time is the same for all the data partitions. In Examples 1 and 2, the INDEX_RECREATE phase is indicated, so the start time is the same for all the data partitions.

Identifying an index rebuild requirement

You can determine if an index rebuild is required by obtaining the value of the maximum reorganize phase element (reorg_max_phase), corresponding to any one of the data partitions being reorganized. If reorg_max_phase has a value of 3 or 4, then an Index Rebuild is required. Examples 1 and 2 report a reorg_max_phase value of 3, indicating an index rebuild is required.

Example

The following sample output is from a three-node server that contains a table with three data partitions:

```
CREATE TABLE  sales (c1 INT, c2 INT, c3 INT)
    PARTITION BY RANGE (c1)
        (PART P1 STARTING FROM (1)  ENDING AT (10) IN parttbs,
         PART P2 STARTING FROM (11) ENDING AT (20) IN parttbs,
         PART P3 STARTING FROM (21) ENDING AT (30) IN parttbs)
    DISTRIBUTE BY (c2)
```

Statement executed:

```
REORG TABLE sales ALLOW NO ACCESS ON ALL DBPARTITIONNUMS
```

Example 1:

```
GET SNAPSHOT FOR TABLES ON DPARTDB GLOBAL
```

The output is modified to include table information for the relevant table only.

```
            Table Snapshot

First database connect timestamp     = 06/28/2005 13:46:43.061690
Last reset timestamp                 = 06/28/2005 13:46:47.440046
Snapshot timestamp                   = 06/28/2005 13:46:50.964033
Database name                        = DPARTDB
Database path                        = /work/sales/NODE0000/SQL00001/
Input database alias                 = DPARTDB
Number of accessed tables            = 5

Table List
 Table Schema        = NEWTON
 Table Name          = SALES
 Table Type          = User
 Data Partition Id   = 0
 Data Object Pages   = 3
 Rows Read           = 12
 Rows Written        = 1
 Overflows           = 0
 Page Reorgs         = 0
 Table Reorg Information:
   Node number       = 0
   Reorg Type        =
        Reclaiming
        Table Reorg
        Allow No Access
        Recluster Via Table Scan
        Reorg Data Only
   Reorg Index       = 0
   Reorg Tablespace  = 3
 Long Temp space ID  = 3
   Start Time        = 06/28/2005 13:46:49.816883
   Reorg Phase       = 3 - Index Recreate
   Max Phase         = 3
   Phase Start Time  = 06/28/2005 13:46:50.362918
```

```
  Status            = Completed
  Current Counter   = 0
  Max Counter       = 0
  Completion        = 0
  End Time          = 06/28/2005 13:46:50.821244

Table Reorg Information:
  Node number       = 1
  Reorg Type        =
       Reclaiming
       Table Reorg
       Allow No Access
       Recluster Via Table Scan
       Reorg Data Only
  Reorg Index       = 0
  Reorg Tablespace  = 3
Long Temp space ID  = 3
  Start Time        = 06/28/2005 13:46:49.822701
  Reorg Phase       = 3 - Index Recreate
  Max Phase         = 3
  Phase Start Time  = 06/28/2005 13:46:50.420741
  Status            = Completed
  Current Counter   = 0
  Max Counter       = 0
  Completion        = 0
  End Time          = 06/28/2005 13:46:50.899543


Table Reorg Information:
  Node number       = 2
  Reorg Type        =
       Reclaiming
       Table Reorg
       Allow No Access
       Recluster Via Table Scan
       Reorg Data Only
  Reorg Index       = 0
  Reorg Tablespace  = 3
Long Temp space ID  = 3
  Start Time        = 06/28/2005 13:46:49.814813
  Reorg Phase       = 3 - Index Recreate
  Max Phase         = 3
  Phase Start Time  = 06/28/2005 13:46:50.344277
  Status            = Completed
  Current Counter   = 0
  Max Counter       = 0
  Completion        = 0
  End Time          = 06/28/2005 13:46:50.803619


Table Schema        = NEWTON
Table Name          = SALES
Table Type          = User
Data Partition Id   = 1
Data Object Pages   = 3
Rows Read           = 8
Rows Written        = 1
Overflows           = 0
Page Reorgs         = 0
Table Reorg Information:
  Node number       = 0
  Reorg Type        =
       Reclaiming
       Table Reorg
       Allow No Access
       Recluster Via Table Scan
       Reorg Data Only
```

```
  Reorg Index       = 0
  Reorg Tablespace  = 3
Long Temp space ID  = 3
  Start Time        = 06/28/2005 13:46:50.014617
  Reorg Phase       = 3 - Index Recreate
  Max Phase         = 3
  Phase Start Time  = 06/28/2005 13:46:50.362918
  Status            = Completed
  Current Counter   = 0
  Max Counter       = 0
  Completion        = 0
  End Time          = 06/28/2005 13:46:50.821244

Table Reorg Information:
  Node number       = 1
  Reorg Type        =
       Reclaiming
       Table Reorg
       Allow No Access
       Recluster Via Table Scan
       Reorg Data Only
  Reorg Index       = 0
  Reorg Tablespace  = 3
Long Temp space ID  = 3
  Start Time        = 06/28/2005 13:46:50.026278
  Reorg Phase       = 3 - Index Recreate
  Max Phase         = 3
  Phase Start Time  = 06/28/2005 13:46:50.420741
  Status            = Completed
  Current Counter   = 0
  Max Counter       = 0
  Completion        = 0
  End Time          = 06/28/2005 13:46:50.899543

Table Reorg Information:
  Node number       = 2
  Reorg Type        =
       Reclaiming
       Table Reorg
       Allow No Access
       Recluster Via Table Scan
       Reorg Data Only
  Reorg Index       = 0
  Reorg Tablespace  = 3
Long Temp space ID  = 3
  Start Time        = 06/28/2005 13:46:50.006392
  Reorg Phase       = 3 - Index Recreate
  Max Phase         = 3
  Phase Start Time  = 06/28/2005 13:46:50.344277
  Status            = Completed
  Current Counter   = 0
  Max Counter       = 0
  Completion        = 0
  End Time          = 06/28/2005 13:46:50.803619


Table Schema        = NEWTON
Table Name          = SALES
Table Type          = User
Data Partition Id   = 2
Data Object Pages   = 3
Rows Read           = 4
Rows Written        = 1
Overflows           = 0
Page Reorgs         = 0
Table Reorg Information:
  Node number       = 0
```

```
  Reorg Type        =
       Reclaiming
       Table Reorg
       Allow No Access
       Recluster Via Table Scan
       Reorg Data Only
  Reorg Index       = 0
  Reorg Tablespace  = 3
Long Temp space ID  = 3
  Start Time        = 06/28/2005 13:46:50.199971
  Reorg Phase       = 3 - Index Recreate
  Max Phase         = 3
  Phase Start Time  = 06/28/2005 13:46:50.362918
  Status            = Completed
  Current Counter   = 0
  Max Counter       = 0
  Completion        = 0
  End Time          = 06/28/2005 13:46:50.821244

Table Reorg Information:
  Node number       = 1
  Reorg Type        =
       Reclaiming
       Table Reorg
       Allow No Access
       Recluster Via Table Scan
       Reorg Data Only
  Reorg Index       = 0
  Reorg Tablespace  = 3
Long Temp space ID  = 3
  Start Time        = 06/28/2005 13:46:50.223742
  Reorg Phase       = 3 - Index Recreate
  Max Phase         = 3
  Phase Start Time  = 06/28/2005 13:46:50.420741
  Status            = Completed
  Current Counter   = 0
  Max Counter       = 0
  Completion        = 0
  End Time          = 06/28/2005 13:46:50.899543

Table Reorg Information:
  Node number       = 2
  Reorg Type        =
       Reclaiming
       Table Reorg
       Allow No Access
       Recluster Via Table Scan
       Reorg Data Only
  Reorg Index       = 0
  Reorg Tablespace  = 3
Long Temp space ID  = 3
  Start Time        = 06/28/2005 13:46:50.179922
  Reorg Phase       = 3 - Index Recreate
  Max Phase         = 3
  Phase Start Time  = 06/28/2005 13:46:50.344277
  Status            = Completed
  Current Counter   = 0
  Max Counter       = 0
  Completion        = 0
  End Time          = 06/28/2005 13:46:50.803619
```

Example 2:

```
GET SNAPSHOT FOR TABLES ON DPARTDB AT DBPARTITIONNUM 2
```

The output is modified to include table information for the relevant table only.

```
            Table Snapshot

First database connect timestamp     = 06/28/2005 13:46:43.617833
Last reset timestamp                 =
Snapshot timestamp                   = 06/28/2005 13:46:51.016787
Database name                        = DPARTDB
Database path                        = /work/sales/NODE0000/SQL00001/
Input database alias                 = DPARTDB
Number of accessed tables            = 3

Table List
 Table Schema        = NEWTON
 Table Name          = SALES
 Table Type          = User
 Data Partition Id   = 0
 Data Object Pages   = 1
 Rows Read           = 0
 Rows Written        = 0
 Overflows           = 0
 Page Reorgs         = 0
 Table Reorg Information:
   Node number       = 2
   Reorg Type        =
        Reclaiming
        Table Reorg
        Allow No Access
        Recluster Via Table Scan
        Reorg Data Only
   Reorg Index       = 0
   Reorg Tablespace  = 3
 Long Temp space ID  = 3
   Start Time        = 06/28/2005 13:46:49.814813
   Reorg Phase       = 3 - Index Recreate
   Max Phase         = 3
   Phase Start Time  = 06/28/2005 13:46:50.344277
   Status            = Completed
   Current Counter   = 0
   Max Counter       = 0
   Completion        = 0
   End Time          = 06/28/2005 13:46:50.803619


 Table Schema        = NEWTON
 Table Name          = SALES
 Table Type          = User
 Data Partition Id   = 1
 Data Object Pages   = 1
 Rows Read           = 0
 Rows Written        = 0
 Overflows           = 0
 Page Reorgs         = 0
 Table Reorg Information:
   Node number       = 2
   Reorg Type        =
        Reclaiming
        Table Reorg
        Allow No Access
        Recluster Via Table Scan
        Reorg Data Only
   Reorg Index       = 0
   Reorg Tablespace  = 3
 Long Temp space ID  = 3
   Start Time        = 06/28/2005 13:46:50.006392
   Reorg Phase       = 3 - Index Recreate
   Max Phase         = 3
   Phase Start Time  = 06/28/2005 13:46:50.344277
   Status            = Completed
```

```
  Current Counter   = 0
  Max Counter       = 0
  Completion        = 0
  End Time          = 06/28/2005 13:46:50.803619


Table Schema        = NEWTON
Table Name          = SALES
Table Type          = User
Data Partition Id   = 2
Data Object Pages   = 1
Rows Read           = 4
Rows Written        = 1
Overflows           = 0
Page Reorgs         = 0
Table Reorg Information:
  Node number       = 2
  Reorg Type        =
       Reclaiming
       Table Reorg
       Allow No Access
       Recluster Via Table Scan
       Reorg Data Only
  Reorg Index       = 0
  Reorg Tablespace  = 3
Long Temp space ID  = 3
  Start Time        = 06/28/2005 13:46:50.179922
  Reorg Phase       = 3 - Index Recreate
  Max Phase         = 3
  Phase Start Time  = 06/28/2005 13:46:50.344277
  Status            = Completed
  Current Counter   = 0
  Max Counter       = 0
  Completion        = 0
  End Time          = 06/28/2005 13:46:50.803619
```

Example 3:

```
SELECT * FROM SYSIBMADM.SNAPLOCK WHERE tabname = 'SALES';
```

The output is modified to include a subset of table information for the relevant table only.

```
...   TBSP_NAME TABNAME LOCK_OBJECT_TYPE   LOCK_MODE  LOCK_STATUS ...
      --------- ------- ------------------ ---------- -----------
...   PARTTBS   SALES      ROW_LOCK        X          GRNT        ...
...   -         SALES      TABLE_LOCK      IX         GRNT        ...
...   PARTTBS   SALES      TABLE_PART_LOCK IX         GRNT        ...
...   PARTTBS   SALES      ROW_LOCK        X          GRNT        ...
...   -         SALES      TABLE_LOCK      IX         GRNT        ...
...   PARTTBS   SALES      TABLE_PART_LOCK IX         GRNT        ...
...   PARTTBS   SALES      ROW_LOCK        X          GRNT        ...
...   -         SALES      TABLE_LOCK      IX         GRNT        ...
...   PARTTBS   SALES      TABLE_PART_LOCK IX         GRNT        ...

 9 record(s) selected.
```

Output from this query (continued).

```
... LOCK_ESCALATION LOCK_ATTRIBUTES DATA_PARTITION_ID DBPARTITIONNUM
    --------------- --------------- ----------------- --------------
...               0 INSERT          2                 2
...               0 NONE            -                 2
...               0 NONE            2                 2
...               0 INSERT          0                 0
...               0 NONE            -                 0
...               0 NONE            0                 0
```

```
...              0 INSERT          1               1
...              0 NONE            -               1
...              0 NONE            1               1
```

Example 4:

```
SELECT * FROM SYSIBMADM.SNAPTAB WHERE tabname = 'SALES';
```

The output is modified to include a subset of table information for the relevant table only.

```
... TABSCHEMA TABNAME TAB_FILE_ID TAB_TYPE   DATA_OBJECT_PAGES ROWS_WRITTEN ...
... --------- ------- ----------- ---------- ----------------- ------------ ...
... NEWTON    SALES             2 USER_TABLE                 1            1 ...
... NEWTON    SALES             4 USER_TABLE                 1            1 ...
... NEWTON    SALES             3 USER_TABLE                 1            1 ...

 3 record(s) selected.
```

Output from this query (continued).

```
... OVERFLOW_ACCESSES PAGE_REORGS DBPARTITIONNUM TBSP_ID DATA_PARTITION_ID
... ----------------- ----------- -------------- ------- -----------------
...                 0           0              0       3                 0
...                 0           0              2       3                 2
...                 0           0              1       3                 1
```

Example 5:

```
SELECT * FROM SYSIBMADM.SNAPTAB_REORG WHERE tabname = 'SALES';;
```

The output is modified to include a subset of table information for the relevant table only.

```
  REORG_PHASE   REORG_MAX_PHASE REORG_TYPE                                      ...
     ------------- -------------- ------------------------------------------
 INDEX_RECREATE              3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
 INDEX_RECREATE              3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
 INDEX_RECREATE              3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
 INDEX_RECREATE              3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
 INDEX_RECREATE              3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
 INDEX_RECREATE              3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
 INDEX_RECREATE              3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
 INDEX_RECREATE              3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...
 INDEX_RECREATE              3  RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY  ...

 9 record(s) selected.
```

Output from this query (continued).

```
... REORG_STATUS REORG_TBSPC_ID DBPARTITIONNUM DATA_PARTITION_ID
    ------------------------- -------------- -----------------
... COMPLETED               3              2                 0
... COMPLETED               3              2                 1
... COMPLETED               3              2                 2
... COMPLETED               3              1                 0
... COMPLETED               3              1                 1
... COMPLETED               3              1                 2
... COMPLETED               3              0                 0
... COMPLETED               3              0                 1
... COMPLETED               3              0                 2
```

Example 6: The Table Reorg Information includes information about reclaiming extents as part of a reorganization operation. The example that follows shows the relevant output.

```
db2 -v "get snapshot for tables on wsdb"

 Table Reorg Information:
   Reorg Type        =
        Reclaim Extents
        Allow Write Access
   Reorg Index       = 0
   Reorg Tablespace  = 0
   Start Time        = 10/22/2008 15:49:35.477532
   Reorg Phase       = 12 - Release
   Max Phase         = 3
```

Note: Any snapshot requests from a monitor version before SQLM_DBMON_VERSION9_7 will not return any Reclaim Reorg status to the requesting client.

Inactive statement tracking for DEADLOCK WITH DETAILS HISTORY event monitors

When running a deadlock event monitor that tracks all statements (and optionally data values) it becomes possible for the system monitor heap to be exhausted by a single application that includes a very high number of statements in a unit of work. It is also possible for the monitor heap to become exhausted if there are a large number of applications executing concurrently.

To mitigate the amount of space that is consumed, inactive statements are written out to the event monitor by an application when the number of inactive statements for that application reaches a certain threshold. After being written to the event monitor, the memory consumed by these inactive statements will be released. Also, if at any time an application is unable to acquire memory from the system monitor heap, that application writes out all of its current inactive statements to the event monitor before trying to acquire the memory again. If the second attempt fails, a message is logged, and the statement history list is truncated for the UOW the application is processing.

The default limit on the number of inactive statements to be kept by any one application is 250. This default value can be overridden using the registry variable DB2_MAX_INACT_STMTS to specify a different value. Users may want to choose a different value for the limit to increase or reduce the amount of system monitor heap used for inactive statement information.

Whenever inactive statements are written out to the event monitor, a message appears in the **db2diag** log file indicating that this has occurred. Whenever the limit for inactive statements has been exceed, a message appears in the **db2diag** log file indicating that this has occurred.

Because an application can now record its statement history entries outside of the context of a deadlock (when one of the thresholds mentioned above are reached), we need a mechanism to associate these entries to the list of statements recorded at the time of a deadlock for analysis. To do this, users can look for statement history entries where:

- `deadlock_id`= 0
- `participant_no` = 0
- `invocation_id`= invocation id of the deadlock
- `application_id`= application identifier of the application that participated in the deadlock

In the case of a write to table event monitor, the number of evmon_activates also needs to be checked.

Notes:

- For SQL statements compiled using the REOPT ALWAYS bind option, there will be no reopt compilation or statement execution data values provided in the deadlock event information.
- At coordinator nodes, when inactive statements are written to the event monitor due to the conditions described in the previous section, the sequence value of all records written will be changed to reflect the current unit of work in process. This is done to help reconcile this data with any data generated later by a deadlock in the same unit of work since all the relevant data can be gathered by searching for the sequence number and application ID information for those records with deadlock_id of 0. This change does mean that the unit of work information is not available for statements started in a preceding unit of work but still active in the current work, as the sequence number will be overwritten by the current unit of work identifier. This behavior does not occur at remote nodes (that is, the original unit of work information is not overwritten) and so care must be taken when trying to reconcile deadlock event records with any records written out prior to the deadlock as the sequence numbers might differ if there are active cursors with hold from previous units of work involved.

Introduction to Windows Management Instrumentation (WMI)

There is an industry initiative that establishes management infrastructure standards and provides a way to combine information from various hardware and software management systems. This initiative is called Web-Based Enterprise Management (WBEM). WBEM is based on the Common Information Model (CIM) schema, which is an industry standard driven by the Desktop Management Task Force (DMTF).

Microsoft Windows Management Instrumentation (WMI) is an implementation of the WBEM initiative for supported Windows platforms. WMI is useful in a Windows enterprise network where it reduces the maintenance and cost of managing enterprise network components. WMI provides:

- A consistent model of Windows operation, configuration, and status.
- A COM API to allow access to management information.
- The ability to operate with other Windows management services.
- A flexible and extensible architecture allowing vendors a means of writing other WMI providers to support new devices, applications, and other enhancements.
- The WMI Query Language (WQL) to create detailed queries of the information.
- An API for management application developers to write Visual Basic or Windows Scripting Host (WSH) scripts.

The WMI architecture has two parts:

1. A management infrastructure that includes the CIM Object Manager (CIMOM) and a central storage area for management data called the CIMOM object repository. CIMOM allows applications to have a uniform way to access management data.
2. WMI providers. WMI providers are the intermediaries between CIMOM and managed objects. Using WMI APIs, WMI providers supply CIMOM with data from managed objects, handle requests on behalf of management applications, and generate event notifications.

Windows Management Instrumentation (WMI) providers are standard COM or DCOM servers that function as mediators between managed objects and the CIM Object Manager (CIMOM). If the CIMOM receives a request from a management application for data that is not available from the CIMOM object repository, or for events, the CIMOM forwards the request to the WMI providers. WMI providers supply data, and event notifications, for managed objects that are specific to their particular domain.

DB2 database system integration with Windows Management Instrumentation

The snapshot monitors can be accessed by Windows Management Instrumentation (WMI) by means of the DB2 performance counters and using the built-in PerfMon provider.

The DB2 profile registry variables can be accessed by WMI by using the built-in Registry provider.

The WMI Software Development Kit (WMI SDK) includes several built-in providers:

- PerfMon provider
- Registry event provider
- Registry provider
- Windows event log provider
- Win32 provider
- WDM provider

The DB2 errors that are in the Event Logs can be accessed by WMI by using the built-in Windows Event Log provider.

DB2 database system has a DB2 WMI Administration provider, and sample WMI script files, to access the following managed objects:

1. Instances of the database server including those instances that are distributed. The following operations can be done:
   - Enumerate instances
   - Configure database manager parameters
   - Start/stop/query the status of the DB2 server service
   - Setup or establish communication
2. Databases. The following operations can be done:
   - Enumerate databases
   - Configure database parameters
   - Create/drop databases
   - Backup/restore/roll forward databases

You will need to register the DB2 WMI provider with the system before running WMI applications. Registration is done by entering the following commands:

- mofcomp %DB2PATH%\bin\db2wmi.mof

  This command loads the definition of the DB2 WMI schema into the system.
- regsvr %DB2PATH%\bin\db2wmi.dll

  This command registers the DB2 WMI provider COM DLL with Windows.

In both commands, %DB2PATH% is the path where DB2 is installed. Also, db2wmi.mof is the .MOF file that contains the DB2 WMI schema definition.

There are several benefits to integrating with the WMI infrastructure:

1. You are able to easily write scripts to manage DB2 servers in a Windows-based environment using the WMI provided tool. Sample Visual Basic (VBS) scripts are provided to carry out simple tasks such as listing instances, creating and dropping databases, and updating configuration parameters. The sample scripts are included in the DB2 Application Development for Windows product.
2. You can create powerful management applications that perform many tasks using WMI. The tasks could include:
   - Displaying system information
   - Monitoring DB2 performance
   - Monitoring DB2 system resource consumption

   By monitoring both system events and DB2 events through this type of management application, you can manage the database better.
3. You can use existing COM and Visual Basic programming knowledge and skills. By providing a COM or Visual Basic interface, your programmers can save time when developing enterprise management applications.

Windows performance monitor introduction

When working with DB2 database manager for Windows, there are tools that can be used to monitor performance.

- **DB2 Performance Expert**

  DB2 Performance Expert for Multiplatforms, Version 1.1 consolidates, reports, analyzes and recommends self-managing and resource tuning changes based on DB2 database performance-related information.
- **IBM InfoSphere® Optim Performance Manager Extended Edition**

  IBM InfoSphere Optim Performance Manager Extended Edition Version 4.1 is a follow on to DB2 Performance Expert. Starting with Version 9.7 Fix Pack 2 and later fix packs, IBM InfoSphere Optim Performance Manager Extended Edition helps optimize the performance and availability of mission critical databases and applications. IBM InfoSphere Optim Performance Manager Extended Edition delivers a proactive, comprehensive performance management approach that allows IT staff to identify, diagnose, solve and prevent performance problems in DB2 products and in associated applications including Java and DB2 Call Level Interface (CLI) applications.
- **DB2 Health Center**

  The functions of the Health Center provide you with different methods to work with performance-related information. These functions somewhat replace the performance monitor capability of the Control Center.

  Important: The Health Center has been deprecated in Version 9.7 and might be removed in a future release. Start using IBM Data Studio and IBM Optim tools. For a mapping between these recommended tools and Control Center tools, see "Table of recommended tools versus Control Center tools" in the *What's New for DB2 Version 9.7* book.
- **Windows Performance Monitor**

  The Windows Performance Monitor enables you to monitor both database and system performance, retrieving information from any of the performance data

providers registered with the system. Windows also provides performance information data on all aspects of computer operation including:

- CPU usage
- Memory utilization
- Disk activity
- Network activity

Registering DB2 with the Windows performance monitor

About this task

The setup program automatically registers DB2 with the Windows Performance Monitor for you.

To make DB2 database and DB2 Connect™ performance information accessible to the Windows Performance Monitor, you must register the DLL for the DB2 for Windows Performance Counters. This also enables any other Windows application using the Win32 performance APIs to get performance data. To install and register the DB2 Performance Counters DLL (DB2Perf.DLL) with the Windows Performance Monitor, type:

```
db2perfi -i
```

Registering the DLL also creates a new key in the services option of the registry. One entry gives the name of the DLL, which provides the counter support. Three other entries give names of functions provided within that DLL. These functions include:

Open Called when the DLL is first loaded by the system in a process.

Collect
Called to request performance information from the DLL.

Close Called when the DLL is unloaded.

Enabling remote access to DB2 performance information

About this task

If your DB2 for Windows workstation is networked to other Windows computers, you can use the feature described in this section.

In order to see Windows performance objects from another DB2 for Windows computer, you must register an administrator username and password with the DB2 database manager. (The default Windows Performance Monitor username, SYSTEM, is a DB2 database reserved word and cannot be used.) To register the name, type:

```
db2perfr -r username password
```

Note: The username used must conform to the DB2 database naming rules.

The username and password data is held in a key in the registry, with security that allows access only by administrators and the SYSTEM account. The data is encoded to prevent security concerns about storing an administrator password in the registry.

Note:

1. Once a username and password combination has been registered with the DB2 database system, even local instances of the Performance Monitor will explicitly log on using that username and password. This means that if the username information registered with DB2 database system does not match, local sessions of the Performance Monitor will not show DB2 database performance information.
2. The username and password combination must be maintained to match the username and password values stored in the Windows Security database. If the username or password is changed in the Windows Security database, the username and password combination used for remote performance monitoring must be reset.
3. To deregister, type:

   ```
   db2perfr -u <username> <password>
   ```

Displaying DB2 database and DB2 Connect performance values

About this task

To display DB2 database and DB2 Connect performance values using the Performance Monitor, simply choose the performance counters whose values you want displayed from the **Add to** box. This box displays a list of performance objects providing performance data. Select an object to see a list of the counters it supplies.

A performance object can also have multiple instances. For example, the LogicalDisk object provides counters such as "% Disk Read Time" and "Disk Bytes/sec"; it also has an instance for each logical drive in the computer, including "C:" and "D:".

Windows performance objects

Windows provides the following performance objects:

- **DB2 Database Manager**

  This object provides general information for a single Windows instance. The DB2 database instance being monitored appears as the object instance.

  For practical and performance reasons, you can only get performance information from one DB2 database instance at a time. The DB2 database instance that the Performance Monitor shows is governed by the `db2instance` registry variable in the Performance Monitor process. If you have multiple DB2 database instances running simultaneously and want to see performance information from more than one, you must start a separate session of the Performance Monitor, with `db2instance` set to the relevant value for each DB2 database instance to be monitored.

  If you are running a partitioned database environment, you can only get performance information from one database partition server at a time. By default, the performance information for the default database partition (that is, the database partition that has logical port 0) is displayed. To see performance information of another database partition, you must start a separate session of the Performance Monitor with the DB2NODE environment variable set to the database partition number of the database partition to be monitored.
- **DB2 Databases**

  This object provides information for a particular database. Information is available for each currently active database.
- **DB2 Applications**

This object provides information for a particular DB2 database application. Information is available for each currently active DB2 database application.

- **DB2 DCS Databases**

  This object provides information for a particular DCS database. Information is available for each currently active database.

- **DB2 DCS Applications**

  This object provides information for a particular DB2 DCS application. Information is available for each currently active DB2 DCS application.

Which of these objects will be listed by the Windows Performance Monitor depends on what is installed on your Windows computer and what applications are active. For example, if the DB2 database manager is installed has been started, the DB2 Database Manager object will be listed. If there are also some DB2 databases and applications currently active on that computer, the DB2 Databases and DB2 Applications objects will be listed as well. If you are using your Windows system as a DB2 Connect gateway and there are some DCS databases and applications currently active, the DB2 DCS Databases and DB2 DCS Applications objects will be listed.

Accessing remote DB2 database performance information

About this task

Enabling remote access to DB2 Performance Information was discussed earlier. In the **Add to** box, select another computer to monitor. This brings up a list of all the available performance objects on that computer.

In order to be able to monitor DB2 Performance object on a remote computer, the level of the DB2 database or DB2 Connect code installed on that computer must be Version 6 or higher.

Resetting DB2 performance values

About this task

When an application calls the DB2 monitor APIs, the information returned is normally the cumulative values since the DB2 database server was started. However, often it is useful to:

- Reset performance values
- Run a test
- Reset the values again
- Re-run the test.

To reset database performance values, use the **db2perfc** program. Type:

```
db2perfc
```

By default, this resets performance values for all active DB2 databases. However, you can also specify a list of databases to reset. You can also use the -d option to specify that performance values for DCS databases should be reset. For example:

```
db2perfc
db2perfc dbalias1 dbalias2 ... dbaliasn

db2perfc  -d
db2perfc  -d  dbalias1 dbalias2 ... dbaliasn
```

The first example resets performance values for all active DB2 databases. The next example resets values for specific DB2 databases. The third example resets performance values for all active DB2 DCS databases. The last example resets values for specific DB2 DCS databases.

The **db2perfc** program resets the values for ALL programs currently accessing database performance information for the relevant DB2 database server instance (that is, the one held in DB2INSTANCE in the session in which you run **db2perfc**.

Invoking **db2perfc** also resets the values seen by anyone remotely accessing DB2 database performance information when the **db2perfc** command is executed.

Note: There is a DB2 database API, **sqlmrset**, that allows an application to reset the values it sees locally, not globally, for particular databases.

Indoubt Transaction Manager overview

Use the Indoubt Transaction Manager window to work with indoubt transactions. The window lists all indoubt transactions for a selected database and one or more selected partitions.

Important: The Indoubt Transaction Manager has been deprecated in Version 9.7 and might be removed in a future release. For more information, see the "Control Center tools have been deprecated" topic in the *What's New for DB2 Version 9.7* book.

An indoubt transaction is a global transaction that was left in an indoubt state. DB2 provides heuristic actions that database administrators can perform on indoubt transactions when the resource owner, such as the database administrator, cannot wait for the Transaction Manager to perform the resync action. This condition may occur if, for example, the communication line is broken, and an indoubt transaction is tying up needed resources such as locks on tables and indexes, log space, and storage used by the transaction.

While it is preferable for the Transaction Manager to initiate the re-sync action, there may be times when you may have to perform the heuristic actions on the indoubt transactions. In these cases, use the heuristic actions with caution and only as a last resort and follow these guidelines.

- The *gtrid* portion of the transaction ID is the global transaction ID that is identical to that in other resource managers (RM) that participate in the global transaction.
- Use your knowledge of the application and the operating environment to identify the other participating resource managers,
- If the transaction manager is CICS®, and the only resource manager is a CICS resource, perform a heuristic rollback.
- If the transaction manager is not CICS, use it to determine the status of the transaction that has the same *gtrid* as the indoubt transaction.
- If, at least, one resource manager has committed or rolled back, perform a heuristic commit or rollback.
- If all the transactions are in the prepared state, perform a heuristic rollback.
- If, at least, one of the resource managers is not available, perform a heuristic rollback.

To open the Indoubt Transaction Manager on Intel platforms, from the **Start** menu, click **Start -→ Programs -→IBM DB2 -→Monitoring Tools -→Indoubt Transaction Manager**.

To open the Indoubt Transaction Manager using the command line in UNIX or on Intel, run the following command:

```
db2indbt
```

You can perform the following heuristic actions on indoubt transactions:

- Forget

  This permits the resource manager to erase knowledge of a heuristically completed transaction by removing the log records and releasing log pages. A heuristically completed transaction is one that has been committed or rolled back heuristically. You can use the forget action on transactions that are heuristically committed or rolled back for a selected database and one or more selected partitions. To forget an indoubt transaction, select a database and partition and then right-click a transaction with a status of **Committed** or **Rolled back** and select **Forget** from the pop-up menu. A confirmation message displays.
- Commit

  This commits an indoubt transaction that is prepared to be committed. If the operation succeeds, the transaction's state becomes heuristically committed. To commit an indoubt transaction, select a database and partition and then right-click a transaction with a status of **Indoubt** or **Missing commit acknowledgement** and select **Commit** from the pop-up menu. A confirmation message displays.
- Rollback

  This rolls back an indoubt transaction that has been prepared. If the operation succeeds, the transaction's state becomes heuristically rolled back. To roll back an indoubt transaction, select a database and partition and then right-click a transaction with a status of **Indoubt** or **Ended** and select **Rollback** from the pop-up menu. A confirmation message displays.

To perform these actions on indoubt transactions you must have SYSADM or DBADM authority.

The columns in the Indoubt Transaction Manager window provide named views that you can use to organize and display indoubt transactions in different ways. The following list describes each of the columns in the interface:

Status
: The indoubt status of the transaction, namely Committed (c), Ended (e), Indoubt (i), Missing commit acknowledgement (m), and Rolled back (r):

  Committed
  : Transactions in this state have been heuristically committed.

  Ended
  : Transactions in this state may have timed out.

  Indoubt
  : Transactions in this state are waiting to be committed or rolled back.

  Missing commit acknowledgement
  : The Transaction Manager is waiting to receive an acknowledgement before committing the transaction.

Rolled back
Transactions in this state have been heuristically rolled back

Timestamp
The time stamp on the server when the transaction entered the prepared (indoubt) state. The time is the local time to the client.

Transaction ID
The XA identifier assigned by the transaction manager to uniquely identify a global transaction.

Application ID
The application identifier assigned by the database manager for this transaction.

Authorization ID
The user ID of the user who ran the transaction.

Sequence Number
The sequence number assigned by the database manager as an extension to the application identifier.

Partition
The partition on which the indoubt transaction exists.

Originator
Indicates whether the transaction was originated by XA or by DB2 in a partitioned database environment.

Log Full
Indicates whether this transaction caused a log full condition.

Type The type information that shows the role of the database in each indoubt transaction.

- **TM** indicates the indoubt transaction is using the database as a transaction manager database.
- **RM** indicates the indoubt transaction is using the database as a resource manager. This means that it is one of the databases participating in the transaction, but is not the transaction manager database.

Part 2. Monitor elements

© Copyright IBM Corp. 1993, 2012

Chapter 11. Monitor elements reported in monitor table functions

DB2 Version 9.7 introduced a number of monitor elements that are reported through new monitor table functions.

These monitor elements provide information about system processing, activities, and data objects, such as tables, table spaces, table space containers, and buffer pools.

Starting with Version 9.7 Fix Pack 2, additional monitor elements provide information about the fast communication manager (FCM).

- "act_aborted_total - Total aborted activities monitor element" on page 494
- "act_completed_total - Total completed activities monitor element" on page 495
- "act_rejected_total - Total rejected activities monitor element" on page 497
- "act_rqsts_total - Total activity requests monitor elements" on page 500
- "activity_id - Activity ID monitor element" on page 502
- "activity_state - Activity state monitor element" on page 504
- "activity_type - Activity type monitor element" on page 504
- "activitytotaltime_threshold_id - Activity total time threshold ID monitor element" on page 505
- "activitytotaltime_threshold_value - Activity total time threshold value monitor element" on page 505
- "activitytotaltime_threshold_violated - Activity total time threshold violated monitor element" on page 506
- "agent_id - Application handle (agent ID) monitor element" on page 506
- "agent_wait_time - Agent wait time monitor element" on page 511
- "agent_waits_total - Total agent waits monitor element" on page 512
- "aggsqltempspace_threshold_id - Aggregate SQL temporary space threshold ID monitor element" on page 517
- "aggsqltempspace_threshold_value - AggSQL temporary space threshold value monitor element" on page 517
- "aggsqltempspace_threshold_violated - AggSQL temporary space threshold violated monitor element" on page 517
- "app_rqsts_completed_total - Total application requests completed monitor element" on page 518
- "appl_id - Application ID monitor element" on page 519
- "appl_name - Application name monitor element" on page 523
- "application_handle - Application handle monitor element" on page 528
- "audit_events_total - Total audit events monitor element" on page 531
- "audit_file_write_wait_time - Audit file write wait time monitor element" on page 532
- "audit_file_writes_total - Total audit files written monitor element" on page 533
- "audit_subsystem_wait_time - Audit subsystem wait time monitor element" on page 534
- "audit_subsystem_waits_total - Total audit subsystem waits monitor element" on page 536

© Copyright IBM Corp. 1993, 2012

- "auto_storage_hybrid - Hybrid automatic storage table space indicator monitor element" on page 539
- "automatic - Buffer pool automatic monitor element" on page 540
- "block_ios - Number of block I/O requests monitor element" on page 541
- "boundary_leaf_node_splits - Boundary leaf node splits monitor element" on page 543
- "bp_name - Buffer pool name monitor element" on page 543
- "buff_auto_tuning - FCM buffer auto-tuning indicator monitor element" on page 544
- "buff_free - FCM buffers currently free monitor element" on page 545
- "buff_free_bottom - Minimum FCM buffers free monitor element" on page 545
- "buff_max - Maximum possible number of FCM buffers monitor element" on page 546
- "buff_total - Number of currently allocated FCM buffers monitor element" on page 546
- "cat_cache_inserts - Catalog cache inserts monitor element" on page 547
- "cat_cache_lookups - Catalog cache lookups monitor element" on page 548
- "ch_auto_tuning - FCM channel auto-tuning indicator monitor element" on page 552
- "ch_free - Channels currently free monitor element" on page 552
- "ch_free_bottom - Minimum channels free monitor element" on page 553
- "ch_max - Maximum possible number of FCM channels monitor element" on page 553
- "ch_total - Number of currently allocated FCM channels monitor element" on page 554
- "client_acctng - Client accounting string monitor element" on page 554
- "client_applname - Client application name monitor element" on page 555
- "client_hostname - Client hostname monitor element" on page 557
- "client_idle_wait_time - Client idle wait time monitor element" on page 557
- "client_pid - Client process ID monitor element" on page 558
- "client_platform - Client operating platform monitor element" on page 559
- "client_port_number - Client port number monitor element" on page 560
- "client_prdid - Client product and version ID monitor element" on page 560
- "client_protocol - Client communication protocol monitor element" on page 561
- "client_userid - Client user ID monitor element" on page 562
- "client_wrkstnname - Client workstation name monitor element" on page 563
- "comp_env_desc - Compilation environment monitor element" on page 565
- "concurrentdbcoordactivities_db_threshold_id - Concurrent database coordinator activities database threshold ID monitor element" on page 569
- "concurrentdbcoordactivities_db_threshold _queued - Concurrent database coordinator activities database threshold queued monitor element" on page 570
- "concurrentdbcoordactivities_db_threshold _value - Concurrent database coordinator activities database threshold value monitor element" on page 570
- "concurrentdbcoordactivities_db_threshold _violated - Concurrent database coordinator activities database threshold violated monitor element" on page 571
- "concurrentdbcoordactivities_subclass_threshold _id - Concurrent database coordinator activities service subclass threshold ID monitor element" on page 571

- "concurrentdbcoordactivities_subclass_ threshold_queued - Concurrent database coordinator activities service subclass threshold queued monitor element" on page 571
- "concurrentdbcoordactivities_subclass_ threshold_value - Concurrent database coordinator activities service subclass threshold value monitor element" on page 572
- "concurrentdbcoordactivities_subclass_ threshold_violated - Concurrent database coordinator activities service subclass threshold violated monitor element" on page 572
- "concurrentdbcoordactivities_superclass_ threshold_id - Concurrent database coordinator activities service superclass threshold ID monitor element" on page 573
- "concurrentdbcoordactivities_superclass_ threshold_queued - Concurrent database coordinator activities service superclass threshold queued monitor element" on page 573
- "concurrentdbcoordactivities_superclass_ threshold_value - Concurrent database coordinator activities service superclass threshold value monitor element" on page 573
- "concurrentdbcoordactivities_superclass_ threshold_violated - Concurrent database coordinator activities service superclass threshold violated monitor element" on page 574
- "concurrentdbcoordactivities_work_action_set_ threshold_id - Concurrent database coordinator activities work action set threshold ID monitor element" on page 576
- "concurrentdbcoordactivities_work_action_set_ threshold_queued - Concurrent database coordinator activities work action set threshold queued monitor element" on page 577
- "concurrentdbcoordactivities_work_action_set_ threshold_value - Concurrent database coordinator activities work action set threshold value monitor element" on page 577
- "concurrentdbcoordactivities_work_action_set_ threshold_violated - Concurrent database coordinator activities work action set threshold violated monitor element" on page 578
- "connection_start_time - Connection start time monitor element" on page 579
- "connection_status - Connection status monitor element" on page 579
- "container_accessible - Accessibility of container monitor element" on page 581
- "container_id - Container identification monitor element" on page 581
- "container_name - Container name monitor element" on page 581
- "container_stripe_set - Container stripe set monitor element" on page 582
- "container_total_pages - Total pages in container monitor element" on page 582
- "container_type - Container type monitor element" on page 583
- "container_usable_pages - Usable pages in container monitor element" on page 583
- "coord_member - Coordinator member monitor element" on page 590
- "cputime_threshold_id - CPU time threshold ID monitor element" on page 598
- "cputime_threshold_value - CPU time threshold value monitor element" on page 599
- "cputime_threshold_violated - CPU time threshold violated monitor element" on page 599

- "cputimeinsc_threshold_id - CPU time in service class threshold ID monitor element" on page 599
- "cputimeinsc_threshold_value - CPU time in service class threshold value monitor element" on page 600
- "cputimeinsc_threshold_violated - CPU time in service class threshold violated monitor element" on page 600
- "current_extent - Extent currently being moved monitor element" on page 603
- "data_partition_id - Data partition identifier monitor element" on page 604
- "db_storage_path_state - Storage path state monitor element" on page 610
- "db_storage_path_with_dpe - Storage path including database partition expression monitor element" on page 611
- "db_work_action_set_id - Database work action set ID monitor element" on page 611
- "db_work_class_id - Database work class ID monitor element" on page 611
- "deadlocks - Deadlocks detected monitor element" on page 614
- "del_keys_cleaned - Pseudo deleted keys cleaned monitor element" on page 617
- "diaglog_write_wait_time - Diagnostic log file write wait time monitor element" on page 618
- "diaglog_writes_total - Total diagnostic log file writes monitor element" on page 619
- "direct_read_reqs - Direct read requests monitor element" on page 620
- "direct_read_time - Direct read time monitor element" on page 622
- "direct_reads - Direct reads from database monitor element" on page 624
- "direct_write_reqs - Direct write requests monitor element" on page 626
- "direct_write_time - Direct write time monitor element" on page 627
- "direct_writes - Direct writes to database monitor element" on page 629
- "eff_stmt_text - Effective statement text monitor element" on page 633
- "effective_isolation - Effective isolation monitor element" on page 633
- "effective_lock_timeout - Effective lock timeout monitor element" on page 634
- "effective_query_degree - Effective query degree monitor element" on page 634
- "empty_pages_deleted - Empty pages deleted monitor element" on page 635
- "empty_pages_reused - Empty pages reused monitor element" on page 635
- "entry_time - Entry time monitor element" on page 636
- "estimatedsqlcost_threshold_id - Estimated SQL cost threshold ID monitor element" on page 636
- "estimatedsqlcost_threshold_value - Estimated SQL cost threshold value monitor element" on page 636
- "estimatedsqlcost_threshold_violated - Estimated SQL cost threshold violated monitor element" on page 637
- "executable_id - Executable ID monitor element" on page 639
- "fcm_message_recv_volume - FCM message received volume monitor element" on page 640
- "fcm_message_recv_wait_time - FCM message received wait time monitor element" on page 641
- "fcm_message_recvs_total - Total FCM message receives monitor element" on page 643
- "fcm_message_send_volume - FCM message send volume monitor element" on page 644

- "fcm_message_send_wait_time - FCM message send wait time monitor element" on page 645
- "fcm_message_sends_total - Total FCM message sends monitor element" on page 646
- "fcm_recv_volume - FCM received volume monitor element" on page 647
- "fcm_recv_wait_time - FCM received wait time monitor element" on page 648
- "fcm_recvs_total - FCM receives total monitor element" on page 650
- "fcm_send_volume - FCM send volume monitor element" on page 651
- "fcm_send_wait_time - FCM send wait time monitor element" on page 652
- "fcm_sends_total - FCM sends total monitor element" on page 654
- "fcm_tq_recv_volume - FCM table queue received volume monitor element" on page 655
- "fcm_tq_recv_wait_time - FCM table queue received wait time monitor element" on page 656
- "fcm_tq_recvs_total - FCM table queue receives total monitor element" on page 657
- "fcm_tq_send_volume - FCM table queue send volume monitor element" on page 658
- "fcm_tq_send_wait_time - FCM table queue send wait time monitor element" on page 659
- "fcm_tq_sends_total - FCM table queue send total monitor element" on page 660
- "files_closed - Database files closed monitor element" on page 662
- "fs_caching - File system caching monitor element" on page 664
- "fs_id - Unique file system identification number monitor element" on page 664
- "fs_total_size - Total size of a file system monitor element" on page 665
- "fs_used_size - Amount of space used on a file system monitor element" on page 665
- "hostname - Host name monitor element" on page 683
- "ida_recv_volume - Total data volume received monitor element" on page 683
- "ida_recv_wait_time - Time spent waiting to receive data monitor element" on page 685
- "ida_recvs_total - Number of times data received monitor element" on page 686
- "ida_send_volume - Total data volume sent monitor element" on page 687
- "ida_send_wait_time - Time spent waiting to send data monitor element" on page 689
- "ida_sends_total - Number of times data sent monitor element" on page 690
- "iid - Index identifier monitor element" on page 692
- "include_col_updates - Include column updates monitor element" on page 693
- "index_only_scans - Index-only scans monitor element" on page 694
- "index_scans - Index scans monitor element" on page 694
- "index_tbsp_id - Index table space ID monitor element" on page 694
- "insert_timestamp - Insert timestamp monitor element" on page 696
- "int_commits - Internal commits monitor element" on page 698
- "int_node_splits - Intermediate node splits monitor element" on page 700
- "int_rollbacks - Internal rollbacks monitor element" on page 700
- "ipc_recv_volume - Interprocess communication received volume monitor element" on page 704

- "ipc_recv_wait_time - Interprocess communication received wait time monitor element" on page 705
- "ipc_recvs_total - Interprocess communication receives total monitor element" on page 706
- "ipc_send_volume - Interprocess communication send volume monitor element" on page 707
- "ipc_send_wait_time - Interprocess communication send wait time monitor element" on page 708
- "ipc_sends_total - Interprocess communication send total monitor element" on page 709
- "key_updates - Key updates monitor element" on page 710
- "last_executable_id - Last executable identifier monitor element" on page 712
- "last_extent - Last extent moved monitor element" on page 712
- "last_reference_time - Last reference time monitor element" on page 713
- "last_request_type - Last request type monitor element" on page 713
- "local_start_time - Local start time monitor element" on page 717
- "lock_escals - Number of lock escalations monitor element" on page 721
- "lock_timeouts - Number of lock timeouts monitor element" on page 732
- "lock_wait_time - Time waited on locks monitor element" on page 734
- "lock_waits - Lock waits monitor element" on page 737
- "locks_held - Locks held monitor element" on page 738
- "log_buffer_wait_time - Log buffer wait time monitor element" on page 740
- "log_disk_wait_time - Log disk wait time monitor element" on page 741
- "log_disk_waits_total - Total log disk waits monitor element" on page 743
- "long_tbsp_id - Long table space ID monitor element" on page 747
- "member - Database member monitor element" on page 760
- "nleaf - Number of leaf pages monitor element" on page 770
- "nlevels - Number of index levels monitor element" on page 770
- "nonboundary_leaf_node_splits - Non-boundary leaf node splits monitor element" on page 771
- "num_exec_with_metrics - Number of executions with metrics collected monitor element" on page 773
- "num_executions - Statement executions monitor element" on page 773
- "num_extents_left - Number of extents left to process monitor element" on page 774
- "num_extents_moved - Number of extents moved monitor element" on page 774
- "num_log_buffer_full - Number of times full log buffer caused agents to wait monitor element" on page 775
- "num_lw_thresh_exceeded - Number of lock wait thresholds exceeded monitor element" on page 778
- "num_remaps - Number of remaps monitor element" on page 780
- "overflow_accesses - Accesses to overflowed records monitor element" on page 789
- "overflow_creates - Overflow creates monitor element" on page 790
- "package_name - Package name monitor element" on page 791
- "package_schema - Package schema monitor element" on page 792
- "package_version_id - Package version monitor element" on page 792

- "page_allocations - Page allocations monitor element" on page 794
- "page_reorgs - Page reorganizations monitor element" on page 794
- "pages_from_block_ios - Total number of pages read by block I/O monitor element" on page 795
- "pages_from_vectored_ios - Total number of pages read by vectored I/O monitor element" on page 796
- "pages_merged - Pages merged monitor element" on page 796
- "pages_read - Number of pages read monitor element" on page 796
- "pages_written - Number of pages written monitor element" on page 796
- "parent_activity_id - Parent activity ID monitor element" on page 797
- "parent_uow_id - Parent unit of work ID monitor element" on page 797
- "pkg_cache_inserts - Package cache inserts monitor element" on page 801
- "pkg_cache_lookups - Package cache lookups monitor element" on page 802
- "pool_async_data_read_reqs - Buffer pool asynchronous read requests monitor element" on page 806
- "pool_async_data_reads - Buffer pool asynchronous data reads monitor element" on page 807
- "pool_async_data_writes - Buffer pool asynchronous data writes monitor element" on page 808
- "pool_async_index_read_reqs - Buffer pool asynchronous index read requests monitor element" on page 808
- "pool_async_index_reads - Buffer pool asynchronous index reads monitor element" on page 809
- "pool_async_index_writes - Buffer pool asynchronous index writes monitor element" on page 810
- "pool_async_xda_read_reqs - Buffer pool asynchronous XDA read requests monitor element" on page 812
- "pool_async_xda_reads - Buffer pool asynchronous XDA data reads monitor element" on page 813
- "pool_async_xda_writes - Buffer pool asynchronous XDA data writes monitor element" on page 814
- "pool_data_l_reads - Buffer pool data logical reads monitor element" on page 816
- "pool_data_p_reads - Buffer pool data physical reads monitor element" on page 818
- "pool_data_writes - Buffer pool data writes monitor element" on page 820
- "pool_drty_pg_steal_clns - Buffer pool victim page cleaners triggered monitor element" on page 822
- "pool_drty_pg_thrsh_clns - Buffer pool threshold cleaners triggered monitor element" on page 823
- "pool_index_l_reads - Buffer pool index logical reads monitor element" on page 825
- "pool_index_p_reads - Buffer pool index physical reads monitor element" on page 827
- "pool_index_writes - Buffer pool index writes monitor element" on page 829
- "pool_lsn_gap_clns - Buffer pool log space cleaners triggered monitor element" on page 831
- "pool_no_victim_buffer - Buffer pool no victim buffers monitor element" on page 832

- "pool_read_time - Total buffer pool physical read time monitor element" on page 833
- "pool_temp_data_l_reads - Buffer pool temporary data logical reads monitor element" on page 835
- "pool_temp_data_p_reads - Buffer pool temporary data physical reads monitor element" on page 837
- "pool_temp_index_l_reads - Buffer pool temporary index logical reads monitor element" on page 839
- "pool_temp_index_p_reads - Buffer pool temporary index physical reads monitor element" on page 841
- "pool_temp_xda_l_reads - Buffer pool temporary XDA data logical reads monitor element" on page 843
- "pool_temp_xda_p_reads - Buffer pool temporary XDA data physical reads monitor element" on page 845
- "pool_write_time - Total buffer pool physical write time monitor element" on page 847
- "pool_xda_l_reads - Buffer pool XDA data logical reads monitor element" on page 849
- "pool_xda_p_reads - Buffer pool XDA data physical reads monitor element" on page 851
- "pool_xda_writes - Buffer pool XDA data writes monitor element" on page 853
- "post_shrthreshold_sorts - Post shared threshold sorts monitor element" on page 856
- "post_threshold_sorts - Post threshold sorts monitor element" on page 858
- "prep_time - Preparation time monitor element" on page 860
- "pseudo_deletes - Pseudo deletes monitor element" on page 868
- "pseudo_empty_pages - Pseudo empty pages monitor element" on page 868
- "qp_query_id - Query patroller query ID monitor element" on page 868
- "query_cost_estimate - Query cost estimate monitor element" on page 869
- "reclaimable_space_enabled - Reclaimable space enabled indicator monitor element" on page 874
- "remote_member - Remote member monitor element" on page 877
- "root_node_splits - Root node splits monitor element" on page 888
- "routine_id - Routine ID monitor element" on page 888
- "rows_deleted - Rows deleted monitor element" on page 889
- "rows_inserted - Rows inserted monitor element" on page 890
- "rows_modified - Rows modified monitor element" on page 890
- "rows_read - Rows read monitor element" on page 892
- "rows_returned - Rows returned monitor element" on page 893
- "rows_updated - Rows updated monitor element" on page 896
- "rqsts_completed_total - Total requests completed monitor element" on page 897
- "sc_work_action_set_id - Service class work action set ID monitor element" on page 898
- "sc_work_class_id - Service class work class ID monitor element" on page 899
- "section_number - Section number monitor element" on page 901
- "section_type - Section type indicator monitor element" on page 902
- "service_class_id - Service class ID monitor element" on page 907

- “service_subclass_name - Service subclass name monitor element” on page 908
- “service_superclass_name - Service superclass name monitor element” on page 909
- “session_auth_id - Session authorization ID monitor element” on page 910
- “sort_overflows - Sort overflows monitor element” on page 915
- “sqlrowsread_threshold_id - SQL rows read threshold ID monitor element” on page 920
- “sqlrowsread_threshold_value - SQL rows read threshold value monitor element” on page 921
- “sqlrowsread_threshold_violated - SQL rows read threshold violated monitor element” on page 921
- “sqlrowsreadinsc_threshold_id - SQL rows read in service class threshold ID monitor element” on page 921
- “sqlrowsreadinsc_threshold_value - SQL rows read in service class threshold value monitor element” on page 922
- “sqlrowsreadinsc_threshold_violated - SQL rows read in service class threshold violated monitor element” on page 922
- “sqlrowsreturned_threshold_id - SQL rows read returned threshold ID monitor element” on page 922
- “sqlrowsreturned_threshold_value - SQL rows read returned threshold value monitor element” on page 923
- “sqlrowsreturned_threshold_violated - SQL rows read returned threshold violated monitor element” on page 923
- “sqltempspace_threshold_id - SQL temporary space threshold ID monitor element” on page 923
- “sqltempspace_threshold_value - SQL temporary space threshold value monitor element” on page 924
- “sqltempspace_threshold_violated - SQL temporary space threshold violated monitor element” on page 924
- “stmt_invocation_id - Statement invocation identifier monitor element” on page 932
- “stmt_nest_level - Statement nesting level monitor element” on page 935
- “stmt_pkgcache_id - Statement package cache identifier monitor element” on page 937
- “stmt_text - SQL statement text monitor element” on page 940
- “system_auth_id - System authorization identifier monitor element” on page 950
- “tab_file_id - Table file ID monitor element” on page 951
- “tab_type - Table type monitor element” on page 951
- “table_file_id - Table file ID monitor element” on page 951
- “table_name - Table name monitor element” on page 952
- “table_scans - Table scans monitor element” on page 953
- “table_schema - Table schema name monitor element” on page 953
- “table_type - Table type monitor element” on page 954
- “tablespace_auto_resize_enabled - Table space automatic resizing enabled monitor element” on page 955
- “tablespace_content_type - Table space content type monitor element” on page 956

- "tablespace_cur_pool_id - Buffer pool currently being used monitor element" on page 956
- "tablespace_extent_size - Table space extent size monitor element" on page 957
- "tablespace_free_pages - Free pages in table space monitor element" on page 957
- "tablespace_id - Table space identification monitor element" on page 958
- "tablespace_name - Table space name monitor element" on page 961
- "tablespace_next_pool_id - Buffer pool that will be used at next startup monitor element" on page 962
- "tablespace_page_size - Table space page size monitor element" on page 963
- "tablespace_page_top - Table space high watermark monitor element" on page 963
- "tablespace_paths_dropped - Table space using dropped path monitor element" on page 964
- "tablespace_pending_free_pages - Pending free pages in table space monitor element" on page 964
- "tablespace_prefetch_size - Table space prefetch size monitor element" on page 965
- "tablespace_rebalancer_mode - Rebalancer mode monitor element" on page 966
- "tablespace_state - Table space state monitor element" on page 969
- "tablespace_total_pages - Total pages in table space monitor element" on page 971
- "tablespace_type - Table space type monitor element" on page 973
- "tablespace_usable_pages - Usable pages in table space monitor element" on page 973
- "tablespace_used_pages - Used pages in table space monitor element" on page 974
- "tablespace_using_auto_storage - Table space enabled for automatic storage monitor element" on page 974
- "tbsp_max_page_top - Maximum table space page high watermark monitor element" on page 975
- "tcpip_recv_volume - TCP/IP received volume monitor element" on page 975
- "tcpip_recv_wait_time - TCP/IP received wait time monitor element" on page 976
- "tcpip_recvs_total - TCP/IP receives total monitor element" on page 977
- "tcpip_send_volume - TCP/IP send volume monitor element" on page 978
- "tcpip_send_wait_time - TCP/IP send wait time monitor element" on page 979
- "tcpip_sends_total - TCP/IP sends total monitor element" on page 980
- "thresh_violations - Number of threshold violations monitor element" on page 981
- "total_act_time - Total activity time monitor element" on page 988
- "total_act_wait_time - Total activity wait time monitor element" on page 989
- "total_app_commits - Total application commits monitor elements" on page 990
- "total_app_rollbacks - Total application rollbacks monitor element" on page 991
- "total_app_rqst_time - Total application request time monitor element" on page 992
- "total_app_section_executions - Total application section executions monitor element" on page 993
- "total_buffers_rcvd - Total FCM buffers received monitor element" on page 994

- "total_buffers_sent - Total FCM buffers sent monitor element" on page 994
- "total_commit_proc_time - Total commits processing time monitor element" on page 995
- "total_commit_time - Total commit time monitor element" on page 996
- "total_compilations - Total compilations monitor element" on page 997
- "total_compile_proc_time - Total compile processing time monitor element" on page 998
- "total_compile_time - Total compile time monitor element" on page 999
- "total_cpu_time - Total CPU time monitor element" on page 1001
- "total_implicit_compilations - Total implicit complications monitor element" on page 1003
- "total_implicit_compile_proc_time - Total implicit compile processing time monitor element" on page 1004
- "total_implicit_compile_time - Total implicit compile time monitor element" on page 1005
- "total_load_proc_time - Total load processing time monitor element" on page 1006
- "total_load_time - Total load time monitor element" on page 1007
- "total_loads - Total loads monitor element" on page 1008
- "total_move_time - Total extent move time monitor element" on page 1010
- "total_reorg_proc_time - Total reorganization processing time monitor element" on page 1011
- "total_reorg_time - Total reorganization time monitor element" on page 1012
- "total_reorgs - Total reorganizations monitor element" on page 1013
- "total_rollback_proc_time - Total rollback processing time monitor element" on page 1014
- "total_rollback_time - Total rollback time monitor element" on page 1015
- "total_routine_user_code_proc_time - Total routine user code processing time monitor element" on page 1020
- "total_routine_user_code_time - Total routine user code time monitor element" on page 1021
- "total_rqst_mapped_in - Total request mapped-in monitor element" on page 1022
- "total_rqst_mapped_out - Total request mapped-out monitor element" on page 1022
- "total_rqst_time - Total request time monitor element" on page 1023
- "total_runstats - Total runtime statistics monitor element" on page 1024
- "total_runstats_proc_time - Total runtime statistics processing time monitor element" on page 1025
- "total_runstats_time - Total runtime statistics time monitor element" on page 1026
- "total_section_sort_proc_time - Total section sort processing time monitor element" on page 1028
- "total_section_sort_time - Total section sort time monitor element" on page 1030
- "total_section_sorts - Total section sorts monitor element" on page 1032
- "total_sorts - Total sorts monitor element" on page 1036
- "total_wait_time - Total wait time monitor element" on page 1038
- "tq_tot_send_spills - Total number of table queue buffers overflowed monitor element" on page 1044

- "unread_prefetch_pages - Unread prefetch pages monitor element" on page 1047
- "uow_id - Unit of work ID monitor element" on page 1048
- "uow_log_space_used - Unit of work log space used monitor element" on page 1049
- "uow_start_time - Unit of work start timestamp monitor element" on page 1050
- "utility_id - Utility ID" on page 1054
- "valid - Section validity indicator monitor element" on page 1056
- "vectored_ios - Number of vectored I/O requests monitor element" on page 1057
- "wlm_queue_assignments_total - Workload manager total queue assignments monitor element" on page 1059
- "wlm_queue_time_total - Workload manager total queue time monitor element" on page 1060
- "workload_id - Workload ID monitor element" on page 1063
- "workload_name - Workload name monitor element" on page 1064
- "workload_occurrence_id - Workload occurrence identifier monitor element" on page 1065
- "workload_occurrence_state - Workload occurrence state monitor element" on page 1066

Chapter 12. Request monitor elements

Use request monitor elements to monitor the database system, specifically the volume of work and the effort expended by the data server to process application requests.

A request is a directive to a database agent to perform some work that expends database resources. Sources of the request can include:

- A directive issued directly by an external application, such as an OPEN or EXECUTE directive. These are referred to as application requests.
- A directive issued by a coordinator agent to a subagent at the same or a different database member.
- A directive issued by an agent at a different database member.

Request monitor elements measure the volume of work or effort expended by the database server to process different types of requests, including overall system processing, requests related to a specific type of processing, and requests related to a specific data server environment.

Some representative monitor elements for measuring overall system processing information are the following:

- The **rqsts_completed_total** monitor element measures the number of completed by the system.
- The **total_rqst_time** monitor element measures the time spent by requests in the data server, including wait time and processing time
- The **total_wait_time** monitor element measures the overall wait time.
- The **total_cpu_time** monitor element measures the CPU usage time.

Some representative monitor elements for measuring client-server processing information are the following:

- The **client_idle_wait_time** monitor element measures the time spent waiting for the next request from an open connection.
- The **tcpip_recv_volume** monitor element measures the volume of data received by the data server from clients over TCP/IP.

Some representative monitor elements for measuring common data server processing operations are the following:

- **pool_data_l_reads** is one of the monitor elements providing information about buffer pool resource usage.
- **pool_read_time** is one of the monitor elements providing information about I/O processing.
- **lock_wait_time** is one of the monitor elements providing information about locks and locking.
- **total_section_sorts** is one of the monitor elements providing information about sorts.

Some representative monitor elements for monitoring processing relevant to selected types of data server environments are the following:

- **fcm_recv_wait_time** is one of the monitor elements measuring fast communications manager (FCM) processing.

© Copyright IBM Corp. 1993, 2012

- **wlm_queue_time_total** is one of the monitor elements measuring workload management control actions.

Accessing request metrics using table functions

You can use the following table functions to access the request metrics:

- MON_GET_SERVICE_SUBCLASS and MON_GET_SERVICE_SUBCLASS_DETAILS
- MON_GET_WORKLOAD and MON_GET_WORKLOAD_DETAILS
- MON_GET_CONNECTION and MON_GET_CONNECTION_DETAILS
- MON_GET_UNIT_OF_WORK and MON_GET_UNIT_OF_WORK_DETAILS

Each table function in this set of monitor table functions has two forms, one of which has a name ending with "DETAILS." The function that does not end with "DETAILS" provides an SQL relational interface that returns the most commonly needed data. The other function provides XML-based access to the monitor data and returns a more comprehensive set of data.

This set of table functions enables you to focus on request metrics at a particular level of aggregation. You can choose the table function that enables you to focus on subset (or aggregation) of the system workload you are interested in a given situation. All of these table functions include a common set of request metric monitor elements. Each table function may return a few additional details not common with all the table functions.

In a database with no user-defined workloads or service classes, all of the user work performed by the database manager occurs in the default user workload and user service class. The table functions that return data for each service class (or workload) return data for a single service class (or workload) that represents the processing for the user workload for the entire database.

In a database with user-defined workloads and service classes, table functions that return data for each service class (or workload) enable you to compare processing per service class (or workload). Using SQL, you can sum the values across service classes (or workloads) to obtain the value of a monitor element that represents the processing for the user workload for the entire database.

Accessing request metrics using event monitors

Request metrics are reported by the following event monitors:

- Statistics event monitor - Request metrics are one of several types of information reported by this event monitor.
- UoW event monitor - This event monitor reports similar or identical fields as the MON_GET_UNIT_OF_WORK table function

Chapter 13. Activity monitor elements

Activity monitor elements are a subset of request monitor elements. Use activity metrics to monitor the subset of data server processing related to executing activities, especially processing done to execute SQL statement sections.

Request monitor elements monitor the complete volume of work and effort expended by the data server to process application requests. Activity monitor elements monitor the work done to execute SQL statement sections, including locking, sorting, and row processing.

To access the current values for activity monitor elements, use the following table functions:

MON_GET_ACTIVITY_DETAILS
: Returns details about one or more activities in progress. Specify the activities of interest in the input parameters. Data returned includes activity metric monitor elements, many other monitor elements, and statement text. Data is returned in XML format.

MON_GET_PKG_CACHE_STMT
: Returns details for some or all SQL statement sections in the database package cache, which includes both static and dynamic SQL statements. Data returned includes activity metric monitor elements aggregated over all executions of the section since it was added to the package cache. Data is returned in a relational form.

Use the activity event monitor to access historical data about activities. This monitor captures data for each execution of each activity. The activity event monitor captures the same activity monitor elements as the MON_GET_ACTIVITY_DETAILS table function. It also captures some additional information.

© Copyright IBM Corp. 1993, 2012

Chapter 14. Data object monitor elements

Data object monitor elements provide information about operations performed on particular data objects, including tables, indexes, buffer pools, table spaces, and containers.

Every data object type has a set of monitor elements that can be monitored. For example, buffer pools have elements that can be used to calculate buffer pool hit ratios.

Use the following table functions to access current values for data object monitor elements. These monitor table functions return data in a relational form:

- MON_GET_BUFFERPOOL
- MON_GET_TABLESPACE
- MON_GET_CONTAINER
- MON_GET_TABLE
- MON_GET_INDEX

© Copyright IBM Corp. 1993, 2012

Chapter 15. Hierarchy of time-spent monitor elements

Many time-spent monitor elements rollup to more general monitor elements. For example, individual wait time elements, such as that for the time spent waiting to receive the next buffer from a table queue (**fcm_tq_recv_wait_time**) and time spent waiting for an FCM reply message (**fcm_message_recv_wait_time**) are both included in overall **fcm_recv_wait_time** element. The hierarchical organization of time-spent monitor element makes it possible to choose the element with the most appropriate level of specificity for your needs.

Dimensions and perspectives for viewing time-spent monitor elements

There are different ways that you can look at the hierarchies of time-spent monitor elements. One way is to look at them from the viewpoint of the system as a whole; you can also look at them in the context of specific activities within the system.

The system-level view or *dimension* includes elements that you can use to see what the system is doing as a whole. You can also use elements in the system dimension to view time-spent information as it relates to specific workloads.

The activity-level view or dimension includes elements that you can use to see where the system is spending time on specific activities, such as the execution of SQL statements. All monitor elements in the activity dimension are included in the higher-level system dimension.

Within each of these two dimensions are two different *perspectives* you can use to look at time-spent monitor elements:

1. Component processing times as compared to wait times
2. Component elapsed times as compared to component processing times.

In first perspective, the values for wait time elements are independent of and complementary to the values for component processing time elements. If you add the sum of all of the reported wait times to the sum of all component processing times, the resulting value would be very close to the value reflected by the **total_rqst_time** monitor element. Any minor difference between the two values is the result of a small amount of miscellaneous component processing time that is not currently tracked in any monitor element.

With the second of these two perspectives, component elapsed times are a superset of component processing times. In other words, for a logical component of the database such as that that performs commits, the total amount of commit processing time (**total_commit_proc_time**) is *included in* the overall elapsed time for the commit (**total_commit_time**). The difference between the total elapsed time and the total processing time is the made up of miscellaneous wait or processing times that are not individually tracked as part of the component elapsed time monitor element.

Viewing component elapsed times relative to wait times is *not* meaningful, because component elapsed times already include wait times incurred as part of the elapsed time in that part of the system. If you were to create a pie chart consisting of both component elapsed times and wait times, it would not be an accurate

© Copyright IBM Corp. 1993, 2012

representation of the time spent in your system because you would actually be double-counting the various types of wait times.

The sections that follow describe the various dimensions (system, activity) and perspectives (component processing/wait times, component elapsed/component processing times) from which you can view time-spent monitor elements.

Note: Not all time-spent elements are reported through all interfaces. For example, the **client_idle_wait_time** monitor element is only applicable to system-level interfaces like the MON_GET_SERVICE_SUBCLASS table function. Refer to the reference topic for each monitor element for a list of the interfaces that report the element.

- "Time-spent monitor element hierarchy - System dimension" on page 242
- "Time-spent monitor element hierarchy - Activity dimension" on page 245

Time-spent monitor element hierarchy - System dimension

Figure 7 on page 243 shows an overall view of how both the wait and component processing time monitor elements relate to one another, as viewed from the system dimension

- "client_idle_wait_time - Client idle wait time monitor element" on page 557
- "total_rqst_time - Total request time monitor element" on page 1023
  - "total_wait_time - Total wait time monitor element" on page 1038
    - "agent_wait_time - Agent wait time monitor element" on page 511
    - "wlm_queue_time_total - Workload manager total queue time monitor element" on page 1060
    - "lock_wait_time - Time waited on locks monitor element" on page 734
    - "log_buffer_wait_time - Log buffer wait time monitor element" on page 740
    - "log_disk_wait_time - Log disk wait time monitor element" on page 741
    - "tcpip_recv_wait_time - TCP/IP received wait time monitor element" on page 976
    - "tcpip_send_wait_time - TCP/IP send wait time monitor element" on page 979
    - "ipc_recv_wait_time - Interprocess communication received wait time monitor element" on page 705
    - "ipc_send_wait_time - Interprocess communication send wait time monitor element" on page 708
    - "fcm_recv_wait_time - FCM received wait time monitor element" on page 648[1]
      - "fcm_tq_recv_wait_time - FCM table queue received wait time monitor element" on page 656[1]
      - "fcm_message_recv_wait_time - FCM message received wait time monitor element" on page 641[1]
    - "fcm_send_wait_time - FCM send wait time monitor element" on page 652[1]
      - "fcm_tq_send_wait_time - FCM table queue send wait time monitor element" on page 659[1]
      - "fcm_message_send_wait_time - FCM message send wait time monitor element" on page 645[1]
    - "audit_subsystem_wait_time - Audit subsystem wait time monitor element" on page 534
    - "audit_file_write_wait_time - Audit file write wait time monitor element" on page 532
    - "diaglog_write_wait_time - Diagnostic log file write wait time monitor element" on page 618
    - "pool_read_time - Total buffer pool physical read time monitor element" on page 833
    - "pool_write_time - Total buffer pool physical write time monitor element" on page 847
    - "direct_read_time - Direct read time monitor element" on page 622
    - "direct_write_time - Direct write time monitor element" on page 627
    - "ida_send_wait_time - Time spent waiting to send data monitor element" on page 689
    - "ida_recv_wait_time - Time spent waiting to receive data monitor element" on page 685
  - "total_compile_proc_time - Total compile processing time monitor element" on page 998
  - "total_implicit_compile_proc_time - Total implicit compile processing time monitor element" on page 1004
  - "total_routine_user_code_proc_time - Total routine user code processing time monitor element" on page 1020
  - "total_section_proc_time - Total section processing time monitor element" on page 1027
    - "total_section_sort_proc_time - Total section sort processing time monitor element" on page 1028
    - *Other*[2]
  - "total_commit_proc_time - Total commits processing time monitor element" on page 995
  - "total_rollback_proc_time - Total rollback processing time monitor element" on page 1014
  - "total_runstats_proc_time - Total runtime statistics processing time monitor element" on page 1025
  - "total_reorg_proc_time - Total reorganization processing time monitor element" on page 1011
  - "total_load_proc_time - Total load processing time monitor element" on page 1006
  - *Other*[3]

[1]These FCM-related wait times do not yield meaningful information when aggregated across members. See "Wait times for FCM communications" on page 246 for more information.

[2]Includes miscellaneous processing and wait times that are not specifically related to this component. For example, **total_section_proc_time** is comprised of **total_section_proc_sort_time** time spent performing activities like table and index scans or joins.

[3] Includes a small number of miscellaneous types of time spent (processing and wait times) that are not currently monitored.

Figure 19. Wait and component processing time spent monitor elements - System dimension. The values for indented monitor elements are included in the element that precedes it in the next-highest level of the hierarchy.

The preceding hierarchy shows how the monitor elements for wait times and component processing times roll-up to the overall request time monitor element (**total_rqst_time**). In other words, the total request time comprises all wait times and all component processing times.

Figure 8 on page 244 shows a detailed view of the monitor elements for time spent in various component areas. Each component time is represented by two different monitor elements:

- One that reports the total amount of *processing time* in a component or stage of processing
- One that reports on the *overall elapsed time* spent in the component. This overall time includes the processing time for the component as well as any miscellaneous other processing or wait times that may be involved.

- "total_rqst_time - Total request time monitor element" on page 1023
  - "total_compile_time - Total compile time monitor element" on page 999
    - "total_compile_proc_time - Total compile processing time monitor element" on page 998
    - *Other*[1]
  - "total_implicit_compile_time - Total implicit compile time monitor element" on page 1005
    - "total_implicit_compile_proc_time - Total implicit compile processing time monitor element" on page 1004
    - *Other*[1]
  - "total_routine_user_code_time - Total routine user code time monitor element" on page 1021
    - "total_routine_user_code_proc_time - Total routine user code processing time monitor element" on page 1020
  - "total_section_time - Total section time monitor element" on page 1033
    - "total_section_sort_time - Total section sort time monitor element" on page 1030
      - "total_section_sort_proc_time - Total section sort processing time monitor element" on page 1028
      - *Other*[1]
    - *Other*[1]
  - "total_commit_time - Total commit time monitor element" on page 996
    - "total_commit_proc_time - Total commits processing time monitor element" on page 995
    - *Other*[1]
  - "total_rollback_time - Total rollback time monitor element" on page 1015
    - "total_rollback_proc_time - Total rollback processing time monitor element" on page 1014
    - *Other*[1]
  - "total_runstats - Total runtime statistics monitor element" on page 1024
    - "total_runstats_proc_time - Total runtime statistics processing time monitor element" on page 1025
    - *Other*[1]
  - "total_reorg_time - Total reorganization time monitor element" on page 1012
    - "total_reorg_proc_time - Total reorganization processing time monitor element" on page 1011
    - *Other*[1]
  - "total_load_time - Total load time monitor element" on page 1007
    - "total_load_proc_time - Total load processing time monitor element" on page 1006
    - *Other*[1]
  - *Other*[2]

[1]Includes one or more different types of wait times.

[2]Includes a small number of miscellaneous types of time spent (processing and wait times) that are not currently monitored.

Figure 20. Component time-spent monitor elements - System dimension. The values for indented monitor elements are included in the element that precedes it in the next-highest level of the hierarchy.

Time-spent monitor element hierarchy - Activity dimension

In addition to being able to view time-spent monitor elements from a system-level perspective, you can also view time-spent monitor elements from the activity dimension. Figure 9 on page 245 shows the monitor elements that you can view for activities from the perspective of wait times as compared to component processing times.

- "stmt_exec_time - Statement execution time monitor element" on page 931
  - "total_act_wait_time - Total activity wait time monitor element" on page 989 [1]
    - "lock_wait_time - Time waited on locks monitor element" on page 734
    - "log_buffer_wait_time - Log buffer wait time monitor element" on page 740
    - "log_disk_wait_time - Log disk wait time monitor element" on page 741
    - "fcm_recv_wait_time - FCM received wait time monitor element" on page 648[2]
      - "fcm_tq_recv_wait_time - FCM table queue received wait time monitor element" on page 656[2]
      - "fcm_message_recv_wait_time - FCM message received wait time monitor element" on page 641[2]
    - "fcm_send_wait_time - FCM send wait time monitor element" on page 652[2]
      - "fcm_tq_send_wait_time - FCM table queue send wait time monitor element" on page 659[2]
      - "fcm_message_send_wait_time - FCM message send wait time monitor element" on page 645[2]
    - "audit_subsystem_wait_time - Audit subsystem wait time monitor element" on page 534
    - "audit_file_write_wait_time - Audit file write wait time monitor element" on page 532
    - "diaglog_write_wait_time - Diagnostic log file write wait time monitor element" on page 618
    - "pool_read_time - Total buffer pool physical read time monitor element" on page 833
    - "pool_write_time - Total buffer pool physical write time monitor element" on page 847
    - "direct_read_time - Direct read time monitor element" on page 622
    - "direct_write_time - Direct write time monitor element" on page 627
    - "ida_send_wait_time - Time spent waiting to send data monitor element" on page 689
    - "ida_recv_wait_time - Time spent waiting to receive data monitor element" on page 685
  - "total_routine_non_sect_proc_time - Non-section processing time monitor element" on page 1017
    - "total_routine_user_code_proc_time - Total routine user code processing time monitor element" on page 1020
    - *Other*[3]
  - "total_section_proc_time - Total section processing time monitor element" on page 1027
    - "total_section_sort_proc_time - Total section sort processing time monitor element" on page 1028
    - *Other*[3]
  - *Other*[4]

[1]Does not include any wait time incurred by nested (child) activities executed by the statement.

[2]These FCM-related wait times do not yield meaningful information when aggregated across members. See "Wait times for FCM communications" on page 246 for more information.

[3]Includes miscellaneous processing times that are not specifically related to this component.

[4]Includes a small number of miscellaneous types of time spent (processing and wait times) that are not currently monitored. In addition, this time includes any processing and wait times incurred by child activities.

Figure 21. Wait and component processing time-spent monitor elements - Activity dimension. The values for indented monitor elements are included in the element that precedes it in the next-highest level of the hierarchy.

Figure 10 on page 246 shows the monitor elements that you can view for activities from the perspective of component elapsed times, which include component processing times.

- "stmt_exec_time - Statement execution time monitor element" on page 931
  - "total_section_time - Total section time monitor element" on page 1033
    - "total_section_sort_time - Total section sort time monitor element" on page 1030
      - "total_section_sort_proc_time - Total section sort processing time monitor element" on page 1028
      - *Other*[1]
    - *Other*[2]
  - "total_routine_time - Total routine time monitor element" on page 1018
    - "total_routine_non_sect_time - Non-section routine execution time monitor elements" on page 1017
      - "total_routine_user_code_time - Total routine user code time monitor element" on page 1021
      - *Other*[2]
    - *Other*[2]

[1]Includes one or more different types of wait times.

[2]Includes miscellaneous processing and wait times that are not specifically related to this component.

Figure 22. Component elapsed and component processing times - Activity dimension. The values for indented monitor elements are included in the element that precedes it in the next-highest level of the hierarchy.

Chapter 16. Logical data groups

Snapshot monitor interface mappings to logical data groups

The following table lists several ways of accessing snapshot monitor data. All snapshot monitor data is stored in monitor elements, which are categorized by logical data groups. Each individual API request type, CLP command, and SQL administrative view only captures monitor data from a subset of all the logical data groups.

Each individual API request type, CLP command, and SQL administrative view listed in this table returns monitor elements from the logical data groups listed in the right-most column.

Note:

1. There are a number of API request types and CLP commands for which there are no corresponding SQL administrative view. For other API request types and CLP commands, individual SQL administrative views capture subsets of the associated logical data groups.
2. Some monitor elements are returned only if the associated monitor switch is set ON. See the individual monitor elements to determine if a required element is under switch control.

Table 58. Snapshot Monitor Interface Mappings to Logical Data Groups

db2GetSnapshot API request type	CLP command	SQL administrative view	Logical data groups
SQLMA_APPLINFO_ALL	list applications [show detail]	APPLICATIONS	appl_info
SQLMA_DBASE_APPLINFO	list applications for database *dbname* [show detail]	APPLICATIONS	appl_info
SQLMA_DCS_APPLINFO_ALL	list dcs applications [show detail]		dcs_appl_info
SQLMA_DB2	get snapshot for dbm	SNAPDBM	db2
		SNAPFCM	fcm
		SNAPFCMPART	fcm_node
		SNAPUTIL	utility_info
		SNAPUTIL_PROGRESS	progress, progress_info
		SNAPDBM_MEMORY_POOL	memory_pool
	get dbm monitor switches	SNAPSWITCHES	switch_list

© Copyright IBM Corp. 1993, 2012

Table 58. Snapshot Monitor Interface Mappings to Logical Data Groups (continued)

db2GetSnapshot API request type	CLP command	SQL administrative view	Logical data groups
SQLMA_DBASE	get snapshot for database on *dbname*	SNAPDB	dbase
		SNAPDETAILLOG	detail_log
		SNAPSTORAGE_PATHS	db_storage_group
			rollforward
			db_sto_path_info
		SNAPTBSP	tablespace
		SNAPDB_MEMORY_POOL	memory_pool
SQLMA_DBASE_ALL	get snapshot for all databases	SNAPDB	dbase
		SNAPSTORAGE_PATHS	db_storage_group
			rollforward
			db_sto_path_info
		SNAPTBSP	tablespace
		SNAPDB_MEMORY_POOL	memory_pool
	list active databases		dbase
SQLMA_DCS_DBASE	get snapshot for dcs database on *dbname*		dcs_dbase, stmt_transmissions
SQLMA_DCS_DBASE_ALL	get snapshot for all dcs databases		dcs_dbase, stmt_transmissions
SQLMA_DBASE_REMOTE	get snapshot for remote database on *dbname*		dbase_remote
SQLMA_DBASE_REMOTE_ALL	get snapshot for all remote databases		dbase_remote
SQLMA_APPL	get snapshot for application applid *appl-id*	SNAPAPPL	appl
		SNAPAGENT	agent
		SNAPAPPL_INFO	appl_info
		SNAPLOCKWAIT	lock_wait
		SNAPSTMT	stmt
		SNAPSUBSECTION	subsection
		SNAPAGENT_MEMORY_POOL	memory_pool
SQLMA_AGENT_ID	get snapshot for application agentid *appl-handle*	SNAPAGENT	appl
		SNAPAGENT	agent
		SNAPAPPL_INFO	appl_info
		SNAPLOCKWAIT	lock_wait
		SNAPSTMT	stmt
		SNAPSUBSECTION	subsection
		SNAPAGENT_MEMORY_POOL	memory pool

Table 58. Snapshot Monitor Interface Mappings to Logical Data Groups (continued)

db2GetSnapshot API request type	CLP command	SQL administrative view	Logical data groups
SQLMA_DBASE_APPLS	get snapshot for applications on *dbname*	SNAPAPPL	appl
		SNAPAGENT	agent
		SNAPAPPL_INFO	appl_info
		SNAPLOCKWAIT	lock_wait
		SNAPSTMT	stmt
		SNAPSUBSECTION	subsection
		SNAPAGENT_MEMORY_POOL	memory_pool
SQLMA_APPL_ALL	get snapshot for all applications	SNAPAPPL	appl
		SNAPAPPL_INFO	appl_info
		SNAPLOCKWAIT	lock_wait
		SNAPSTATEMENT	stmt
		SNAPAGENT	agent
		SNAPSUBSECTION	subsection
		SNAPAGENT_MEMORY_POOL	memory_pool
SQLMA_DCS_APPL	get snapshot for dcs application applid *appl-id*		dcs_appl, dcs_stmt, dcs_appl_info, stmt_transmissions
SQLMA_DCS_APPL_ALL	get snapshot for all dcs applications		dcs_appl, dcs_stmt, dcs_appl_info, stmt_transmissions
SQLMA_DCS_APPL_HANDLE	get snapshot for dcs application agentid *appl-handle*		dcs_appl, dcs_stmt, dcs_appl_info, stmt_transmissions
SQLMA_DCS_DBASE_APPLS	get snapshot for dcs applications on *dbname*		dcs_appl, dcs_stmt, dcs_appl_info, stmt_transmissions
SQLMA_DBASE_APPLS_REMOTE	get snapshot for remote applications on *dbname*		dbase_appl
SQLMA_APPL_REMOTE_ALL	get snapshot for all remote applications		dbase_appl
SQLMA_DBASE_TABLES	get snapshot for tables on *dbname*	SNAPTAB	table
		SNAPTAB_REORG	table_reorg
			table_list
SQLMA_APPL_LOCKS	get snapshot for locks for application applid *appl-id*	SNAPLOCK, SNAPAPPL, SNAPLOCKWAIT	appl_lock_list, lock_wait, lock
SQLMA_APPL_LOCKS_AGENT_ID	get snapshot for locks for application agentid *appl-handle*	SNAPLOCK, SNAPAPPL, SNAPLOCKWAIT	appl_lock_list, lock_wait, lock

Table 58. Snapshot Monitor Interface Mappings to Logical Data Groups (continued)

db2GetSnapshot API request type	CLP command	SQL administrative view	Logical data groups
SQLMA_DBASE_LOCKS	get snapshot for locks on *dbname*	SNAPLOCK	appl_lock_list, lock
		SNAPLOCK, SNAPLOCKWAIT	db_lock_list, lock_wait
SQLMA_DBASE_TABLESPACES	get snapshot for tablespaces on *dbname*	SNAPTBSP	tablespace
		SNAPTBSPPART	tablespace, tablespace_nodeinfo
		SNAPTBSP_QUIESCER	tablespace_quiescer, tablespace_nodeinfo
		SNAPCONTAINER	tablespace_container, tablespace_nodeinfo
		SNAPTBSP_RANGE	tablespace_ranges, tablespace_nodeinfo
			tablespace_list, tablespace_nodeinfo
SQLMA_BUFFERPOOLS_ALL	get snapshot for all bufferpools	SNAPBP	bufferpool
SQLMA_DBASE_BUFFERPOOLS	get snapshot for bufferpools on *dbname*	SNAPBP	bufferpool
SQLMA_DYNAMIC_SQL	get snapshot for dynamic sql on *dbname*	SNAPDYN_SQL	dynsql
			dynsql_list

The following figure shows the order that logical data groupings may appear in a snapshot data stream.

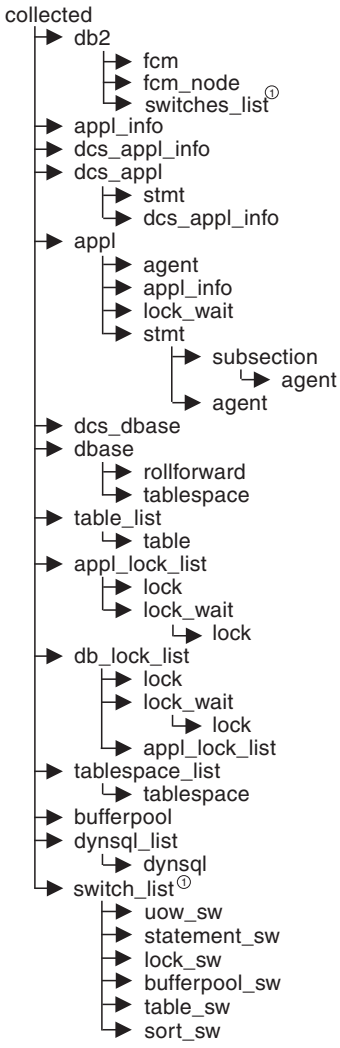


Figure 23. Data Stream Hierarchy

Note: Times may be returned as part of any logical data grouping.

Snapshot monitor logical data groups and monitor elements

The following sections list the logical data groupings and monitor elements that can be returned by snapshot monitoring.

- "agent logical data group" on page 424
- "appl logical data group" on page 424
- "appl_id_info logical data group" on page 427
- "appl_info logical data group" on page 428
- "appl_lock_list logical data group" on page 428
- "appl_remote logical data group" on page 429
- "bufferpool logical data group" on page 429
- "bufferpool_nodeinfo logical data group" on page 431
- "collected logical data group" on page 431
- "db2 logical data group" on page 431

- “db_lock_list logical data group” on page 432
- “dbase logical data group” on page 432
- “dbase_remote logical data group” on page 437
- “db_storage_group logical data group” on page 437
- “dcs_appl logical data group” on page 438
- “dcs_appl_info logical data group” on page 439
- “dcs_dbase logical data group” on page 440
- “dcs_stmt logical data group” on page 442
- “detail_log logical data group” on page 442
- “dynsql logical data group” on page 442
- “dynsql_list logical data group” on page 444
- “fcm logical data group” on page 444
- “fcm_node logical data group” on page 444
- “hadr logical data group” on page 444
- “lock logical data group” on page 445
- “lock_wait logical data group” on page 445
- “memory_pool logical data group” on page 446
- “progress logical data group” on page 446
- “progress_list logical data group” on page 446
- “rollforward logical data group” on page 446
- “stmt logical data group” on page 446
- “stmt_transmissions logical data group” on page 447
- “subsection logical data group” on page 449
- “table logical data group” on page 449
- “table_list logical data group” on page 450
- “table_reorg logical data group” on page 450
- “tablespace logical data group” on page 450
- “tablespace_container logical data group” on page 452
- “tablespace_list logical data group” on page 452
- “tablespace_nodeinfo logical data group” on page 452
- “tablespace_quiescer logical data group” on page 453
- “tablespace_range logical data group” on page 454
- “utility_info logical data group” on page 454

agent logical data group

“agent_pid - Engine dispatchable unit (EDU) identifier monitor element” on page 509

“lock_timeout_val - Lock timeout value monitor element” on page 731

appl logical data group

“acc_curs_blk - Accepted Block Cursor Requests” on page 494

“agent_sys_cpu_time - System CPU Time used by Agent” on page 509

“agent_usr_cpu_time - User CPU Time used by Agent” on page 510

“agents_stolen - Stolen Agents” on page 514

“appl_con_time - Connection Request Start Timestamp” on page 519

“appl_idle_time - Application Idle Time” on page 522

"appl_priority - Application Agent Priority" on page 523
"appl_priority_type - Application Priority Type" on page 524
"associated_agents_top - Maximum Number of Associated Agents" on page 530
"authority_bitmap - User authorization level monitor element" on page 537
"authority_lvl - User authorization level monitor element" on page 538
"binds_precompiles - Binds/Precompiles Attempted" on page 540
"cat_cache_inserts - Catalog cache inserts monitor element" on page 547
"cat_cache_lookups - Catalog cache lookups monitor element" on page 548
"cat_cache_overflows - Catalog Cache Overflows" on page 550
"commit_sql_stmts - Commit Statements Attempted" on page 565
"conn_complete_time - Connection Request Completion Timestamp" on page 578
"ddl_sql_stmts - Data Definition Language (DDL) SQL Statements" on page 613
"deadlocks - Deadlocks detected monitor element" on page 614
"direct_read_reqs - Direct read requests monitor element" on page 620
"direct_read_time - Direct read time monitor element" on page 622
"direct_reads - Direct reads from database monitor element" on page 624
"direct_write_reqs - Direct write requests monitor element" on page 626
"direct_write_time - Direct write time monitor element" on page 627
"direct_writes - Direct writes to database monitor element" on page 629
"dynamic_sql_stmts - Dynamic SQL Statements Attempted" on page 632
"failed_sql_stmts - Failed Statement Operations" on page 640
"hash_join_overflows - Hash Join Overflows" on page 678
"hash_join_small_overflows - Hash Join Small Overflows" on page 679
"inbound_comm_address - Inbound Communication Address" on page 693
"int_auto_rebinds - Internal Automatic Rebinds" on page 697
"int_commits - Internal commits monitor element" on page 698
"int_deadlock_rollbacks - Internal Rollbacks Due To Deadlock" on page 699
"int_rollbacks - Internal rollbacks monitor element" on page 700
"int_rows_deleted - Internal Rows Deleted" on page 702
"int_rows_inserted - Internal Rows Inserted" on page 702
"int_rows_updated - Internal Rows Updated" on page 703
"last_reset - Last Reset Timestamp" on page 714
"lock_escalation - Lock escalation monitor element" on page 720
"lock_timeout_val - Lock timeout value monitor element" on page 731
"lock_timeouts - Number of lock timeouts monitor element" on page 732
"lock_wait_time - Time waited on locks monitor element" on page 734
"lock_waits - Lock waits monitor element" on page 737
"locks_held - Locks held monitor element" on page 738
"locks_waiting - Current agents waiting on locks monitor element" on page 740
"num_agents - Number of Agents Working on a Statement" on page 771
"olap_func_overflows - OLAP Function Overflows monitor element" on page 782
"open_loc_curs - Open Local Cursors" on page 783
"open_loc_curs_blk - Open Local Cursors with Blocking" on page 783

"open_rem_curs - Open Remote Cursors" on page 784
"open_rem_curs_blk - Open Remote Cursors with Blocking" on page 784
"pkg_cache_inserts - Package cache inserts monitor element" on page 801
"pkg_cache_lookups - Package cache lookups monitor element" on page 802
"pool_data_l_reads - Buffer pool data logical reads monitor element" on page 816
"pool_data_p_reads - Buffer pool data physical reads monitor element" on page 818
"pool_data_writes - Buffer pool data writes monitor element" on page 820
"pool_index_l_reads - Buffer pool index logical reads monitor element" on page 825
"pool_index_p_reads - Buffer pool index physical reads monitor element" on page 827
"pool_index_writes - Buffer pool index writes monitor element" on page 829
"pool_read_time - Total buffer pool physical read time monitor element" on page 833
"pool_temp_data_l_reads - Buffer pool temporary data logical reads monitor element" on page 835
"pool_temp_data_p_reads - Buffer pool temporary data physical reads monitor element" on page 837
"pool_temp_index_l_reads - Buffer pool temporary index logical reads monitor element" on page 839
"pool_temp_index_p_reads - Buffer pool temporary index physical reads monitor element" on page 841
"pool_temp_xda_l_reads - Buffer pool temporary XDA data logical reads monitor element" on page 843
"pool_temp_xda_p_reads - Buffer pool temporary XDA data physical reads monitor element" on page 845
"pool_write_time - Total buffer pool physical write time monitor element" on page 847
"pool_xda_l_reads - Buffer pool XDA data logical reads monitor element" on page 849
"pool_xda_p_reads - Buffer pool XDA data physical reads monitor element" on page 851
"pool_xda_writes - Buffer pool XDA data writes monitor element" on page 853
"prefetch_wait_time - Time waited for prefetch monitor element" on page 860
"prev_uow_stop_time - Previous Unit of Work Completion Timestamp" on page 861
"priv_workspace_num_overflows - Private Workspace Overflows" on page 862
"priv_workspace_section_inserts - Private Workspace Section Inserts" on page 863
"priv_workspace_section_lookups - Private Workspace Section Lookups" on page 863
"priv_workspace_size_top - Maximum Private Workspace Size" on page 864
"rej_curs_blk - Rejected Block Cursor Requests" on page 875
"rollback_sql_stmts - Rollback Statements Attempted" on page 886
"rows_deleted - Rows deleted monitor element" on page 889
"rows_inserted - Rows inserted monitor element" on page 890

"rows_read - Rows read monitor element" on page 892
"rows_selected - Rows Selected" on page 895
"rows_updated - Rows updated monitor element" on page 896
"rows_written - Rows Written" on page 897
"select_sql_stmts - Select SQL Statements Executed" on page 902
"shr_workspace_num_overflows - Shared Workspace Overflows" on page 911
"shr_workspace_section_inserts - Shared Workspace Section Inserts" on page 911
"shr_workspace_section_lookups - Shared Workspace Section Lookups" on page 912
"shr_workspace_size_top - Maximum Shared Workspace Size" on page 912
"sort_overflows - Sort overflows monitor element" on page 915
"sql_reqs_since_commit - SQL Requests Since Last Commit" on page 919
"static_sql_stmts - Static SQL Statements Attempted" on page 927
"total_hash_joins - Total Hash Joins" on page 1002
"total_hash_loops - Total Hash Loops" on page 1003
"total_olap_funcs - Total OLAP Functions monitor element" on page 1010
"total_sort_time - Total sort time monitor element" on page 1035
"total_sorts - Total sorts monitor element" on page 1036
"uid_sql_stmts - Update/Insert/Delete SQL Statements Executed" on page 1046
"unread_prefetch_pages - Unread prefetch pages monitor element" on page 1047
"uow_comp_status - Unit of Work Completion Status" on page 1047
"uow_elapsed_time - Most Recent Unit of Work Elapsed Time" on page 1048
"uow_lock_wait_time - Total time unit of work waited on locks monitor element" on page 1049
"uow_log_space_used - Unit of work log space used monitor element" on page 1049
"uow_start_time - Unit of work start timestamp monitor element" on page 1050
"uow_stop_time - Unit of work stop timestamp monitor element" on page 1051
"x_lock_escals - Exclusive lock escalations monitor element" on page 1066
"xquery_stmts - XQuery Statements Attempted" on page 1068

appl_id_info logical data group

"agent_id - Application handle (agent ID) monitor element" on page 506
"appl_id - Application ID monitor element" on page 519
"appl_name - Application name monitor element" on page 523
"appl_status - Application Status" on page 525
"auth_id - Authorization ID" on page 537
"client_db_alias - Database Alias Used by Application" on page 556
"client_prdid - Client product and version ID monitor element" on page 560
"codepage_id - ID of Code Page Used by Application" on page 564
"db_name - Database Name" on page 608
"db_path - Database Path" on page 609
"input_db_alias - Input Database Alias" on page 695
"sequence_no - Sequence number monitor element" on page 904

"status_change_time - Application Status Change Time" on page 930

appl_info logical data group

"agent_id - Application handle (agent ID) monitor element" on page 506
"appl_id - Application ID monitor element" on page 519
"appl_name - Application name monitor element" on page 523
"appl_section_inserts - Section Inserts monitor element" on page 525
"appl_section_lookups - Section Lookups" on page 525
"appl_status - Application Status" on page 525
"auth_id - Authorization ID" on page 537
"authority_bitmap - User authorization level monitor element" on page 537
"authority_lvl - User authorization level monitor element" on page 538
"client_db_alias - Database Alias Used by Application" on page 556
"client_pid - Client process ID monitor element" on page 558
"client_platform - Client operating platform monitor element" on page 559
"client_prdid - Client product and version ID monitor element" on page 560
"client_protocol - Client communication protocol monitor element" on page 561
"codepage_id - ID of Code Page Used by Application" on page 564
"coord_agent_pid - Coordinator agent identifier monitor element" on page 589
"coord_node - Coordinating Node" on page 590
"corr_token - DRDA Correlation Token" on page 591
"db_name - Database Name" on page 608
"db_path - Database Path" on page 609
"execution_id - User Login ID" on page 639
"input_db_alias - Input Database Alias" on page 695
"is_system_appl - Is System Application monitor element" on page 710
"num_assoc_agents - Number of Associated Agents" on page 771
"sequence_no - Sequence number monitor element" on page 904
"session_auth_id - Session authorization ID monitor element" on page 910
"status_change_time - Application Status Change Time" on page 930
"territory_code - Database Territory Code" on page 981
"tpmon_acc_str - TP monitor client accounting string monitor element" on page 1039
"tpmon_client_app - TP monitor client application name monitor element" on page 1040
"tpmon_client_userid - TP monitor client user ID monitor element" on page 1040
"tpmon_client_wkstn - TP monitor client workstation name monitor element" on page 1041
"workload_id - Workload ID monitor element" on page 1063

appl_lock_list logical data group

"agent_id - Application handle (agent ID) monitor element" on page 506
"appl_id - Application ID monitor element" on page 519
"appl_name - Application name monitor element" on page 523
"appl_status - Application Status" on page 525

"auth_id - Authorization ID" on page 537
"client_db_alias - Database Alias Used by Application" on page 556
"codepage_id - ID of Code Page Used by Application" on page 564
"lock_wait_time - Time waited on locks monitor element" on page 734
"locks_held - Locks held monitor element" on page 738
"locks_waiting - Current agents waiting on locks monitor element" on page 740
"sequence_no - Sequence number monitor element" on page 904
"session_auth_id - Session authorization ID monitor element" on page 910
"status_change_time - Application Status Change Time" on page 930

appl_remote logical data group

"commit_sql_stmts - Commit Statements Attempted" on page 565
"create_nickname - Create Nicknames" on page 600
"create_nickname_time - Create Nickname Response Time" on page 601
"datasource_name - Data Source Name" on page 605
"db_name - Database Name" on page 608
"delete_sql_stmts - Deletes" on page 617
"delete_time - Delete Response Time" on page 617
"failed_sql_stmts - Failed Statement Operations" on page 640
"insert_sql_stmts - Inserts" on page 695
"insert_time - Insert Response Time" on page 696
"passthru_time - Pass-Through Time" on page 799
"passthrus - Pass-Through" on page 800
"remote_lock_time - Remote Lock Time" on page 876
"remote_locks - Remote Locks" on page 877
"rollback_sql_stmts - Rollback Statements Attempted" on page 886
"rows_deleted - Rows deleted monitor element" on page 889
"rows_inserted - Rows inserted monitor element" on page 890
"rows_selected - Rows Selected" on page 895
"rows_updated - Rows updated monitor element" on page 896
"select_sql_stmts - Select SQL Statements Executed" on page 902
"select_time - Query Response Time" on page 903
"sp_rows_selected - Rows Returned by Stored Procedures" on page 918
"stored_proc_time - Stored Procedure Time" on page 947
"stored_procs - Stored Procedures" on page 947
"update_sql_stmts - Updates" on page 1052
"update_time - Update Response Time" on page 1053

bufferpool logical data group

"block_ios - Number of block I/O requests monitor element" on page 541
"bp_id - Buffer pool identifier monitor element" on page 543
"bp_name - Buffer pool name monitor element" on page 543
"db_name - Database Name" on page 608
"db_path - Database Path" on page 609
"direct_read_reqs - Direct read requests monitor element" on page 620
"direct_read_time - Direct read time monitor element" on page 622

"direct_reads - Direct reads from database monitor element" on page 624
"direct_write_reqs - Direct write requests monitor element" on page 626
"direct_write_time - Direct write time monitor element" on page 627
"direct_writes - Direct writes to database monitor element" on page 629
"files_closed - Database files closed monitor element" on page 662
"input_db_alias - Input Database Alias" on page 695
"pages_from_block_ios - Total number of pages read by block I/O monitor element" on page 795
"pages_from_vectored_ios - Total number of pages read by vectored I/O monitor element" on page 796
"pool_async_data_read_reqs - Buffer pool asynchronous read requests monitor element" on page 806
"pool_async_data_reads - Buffer pool asynchronous data reads monitor element" on page 807
"pool_async_data_writes - Buffer pool asynchronous data writes monitor element" on page 808
"pool_async_index_read_reqs - Buffer pool asynchronous index read requests monitor element" on page 808
"pool_async_index_reads - Buffer pool asynchronous index reads monitor element" on page 809
"pool_async_index_writes - Buffer pool asynchronous index writes monitor element" on page 810
"pool_async_read_time - Buffer Pool Asynchronous Read Time" on page 811
"pool_async_write_time - Buffer pool asynchronous write time monitor element" on page 812
"pool_async_xda_read_reqs - Buffer pool asynchronous XDA read requests monitor element" on page 812
"pool_async_xda_reads - Buffer pool asynchronous XDA data reads monitor element" on page 813
"pool_async_xda_writes - Buffer pool asynchronous XDA data writes monitor element" on page 814
"pool_data_l_reads - Buffer pool data logical reads monitor element" on page 816
"pool_data_p_reads - Buffer pool data physical reads monitor element" on page 818
"pool_data_writes - Buffer pool data writes monitor element" on page 820
"pool_index_l_reads - Buffer pool index logical reads monitor element" on page 825
"pool_index_p_reads - Buffer pool index physical reads monitor element" on page 827
"pool_index_writes - Buffer pool index writes monitor element" on page 829
"pool_no_victim_buffer - Buffer pool no victim buffers monitor element" on page 832
"pool_read_time - Total buffer pool physical read time monitor element" on page 833
"pool_temp_data_l_reads - Buffer pool temporary data logical reads monitor element" on page 835
"pool_temp_data_p_reads - Buffer pool temporary data physical reads monitor element" on page 837

"pool_temp_index_l_reads - Buffer pool temporary index logical reads monitor element" on page 839
"pool_temp_index_p_reads - Buffer pool temporary index physical reads monitor element" on page 841
"pool_temp_xda_l_reads - Buffer pool temporary XDA data logical reads monitor element" on page 843
"pool_temp_xda_p_reads - Buffer pool temporary XDA data physical reads monitor element" on page 845
"pool_write_time - Total buffer pool physical write time monitor element" on page 847
"pool_xda_l_reads - Buffer pool XDA data logical reads monitor element" on page 849
"pool_xda_p_reads - Buffer pool XDA data physical reads monitor element" on page 851
"pool_xda_writes - Buffer pool XDA data writes monitor element" on page 853
"vectored_ios - Number of vectored I/O requests monitor element" on page 1057

bufferpool_nodeinfo logical data group

"bp_cur_buffsz - Current Size of Buffer Pool" on page 543
"bp_new_buffsz - New Buffer Pool Size" on page 544
"bp_pages_left_to_remove - Number of Pages Left to Remove" on page 544
"bp_tbsp_use_count - Number of Table Spaces Mapped to Buffer Pool" on page 544
"node_number - Node Number" on page 770

collected logical data group

"node_number - Node Number" on page 770
"server_db2_type - Database Manager Type at Monitored (Server) Node" on page 905
"server_instance_name - Server Instance Name" on page 905
"server_prdid - Server Product/Version ID" on page 906
"server_version - Server Version" on page 907
"time_stamp - Snapshot Time" on page 986
"time_zone_disp - Time Zone Displacement" on page 987

db2 logical data group

"agents_created_empty_pool - Agents Created Due to Empty Agent Pool" on page 513
"agents_from_pool - Agents Assigned From Pool" on page 513
"agents_registered - Agents Registered" on page 514
"agents_registered_top - Maximum Number of Agents Registered" on page 514
"agents_stolen - Stolen Agents" on page 514
"agents_waiting_on_token - Agents Waiting for a Token" on page 515
"agents_waiting_top - Maximum Number of Agents Waiting monitor element" on page 516
"comm_private_mem - Committed Private Memory" on page 564
"con_local_dbases - Local Databases with Current Connects" on page 567

"coord_agents_top - Maximum Number of Coordinating Agents" on page 589
"db2start_time - Start Database Manager Timestamp" on page 606
"db_status - Status of Database" on page 609
"gw_cons_wait_client - Number of Connections Waiting for the Client to Send Request" on page 667
"gw_cons_wait_host - Number of Connections Waiting for the Host to Reply" on page 667
"gw_cur_cons - Current Number of Connections for DB2 Connect" on page 668
"gw_total_cons - Total Number of Attempted Connections for DB2 Connect" on page 669
"idle_agents - Number of Idle Agents" on page 692
"last_reset - Last Reset Timestamp" on page 714
"local_cons - Local Connections" on page 715
"local_cons_in_exec - Local Connections Executing in the Database Manager" on page 716
"max_agent_overflows - Maximum Agent Overflows" on page 748
"num_gw_conn_switches - Connection Switches" on page 775
"num_nodes_in_db2_instance - Number of Nodes in Partition" on page 780
"piped_sorts_accepted - Piped Sorts Accepted" on page 800
"piped_sorts_requested - Piped Sorts Requested" on page 801
"post_threshold_hash_joins - Hash Join Threshold" on page 857
"post_threshold_olap_funcs - OLAP Function Threshold monitor element" on page 858
"post_threshold_sorts - Post threshold sorts monitor element" on page 858
"product_name - Product Name" on page 865
"rem_cons_in - Remote Connections To Database Manager" on page 875
"rem_cons_in_exec - Remote Connections Executing in the Database Manager" on page 876
"service_level - Service Level" on page 908
"smallest_log_avail_node - Node with Least Available Log Space" on page 913
"sort_heap_allocated - Total Sort Heap Allocated" on page 914
"sort_heap_top - Sort private heap high watermark" on page 914

db_lock_list logical data group

"appls_cur_cons - Applications Connected Currently" on page 529
"db_name - Database Name" on page 608
"db_path - Database Path" on page 609
"input_db_alias - Input Database Alias" on page 695
"locks_held - Locks held monitor element" on page 738
"locks_waiting - Current agents waiting on locks monitor element" on page 740

dbase logical data group

"active_hash_joins - Active hash joins" on page 501
"active_olap_funcs - Active OLAP Functions monitor element" on page 501
"active_sorts - Active Sorts" on page 502
"agents_top - Number of Agents Created" on page 515
"appl_id_oldest_xact - Application with Oldest Transaction" on page 522

"appl_section_inserts - Section Inserts monitor element" on page 525
"appl_section_lookups - Section Lookups" on page 525
"appls_cur_cons - Applications Connected Currently" on page 529
"appls_in_db2 - Applications Executing in the Database Currently" on page 529
"async_runstats – Total number of asynchronous RUNSTATS requests monitor element" on page 530
"binds_precompiles - Binds/Precompiles Attempted" on page 540
"blocks_pending_cleanup - Pending cleanup rolled-out blocks monitor element" on page 542
"cat_cache_inserts - Catalog cache inserts monitor element" on page 547
"cat_cache_lookups - Catalog cache lookups monitor element" on page 548
"cat_cache_overflows - Catalog Cache Overflows" on page 550
"cat_cache_size_top - Catalog cache high watermark monitor element" on page 551
"catalog_node - Catalog Node Number" on page 551
"catalog_node_name - Catalog Node Network Name" on page 552
"commit_sql_stmts - Commit Statements Attempted" on page 565
"connections_top - Maximum Number of Concurrent Connections" on page 580
"coord_agents_top - Maximum Number of Coordinating Agents" on page 589
"db_conn_time - Database activation timestamp monitor element" on page 607
"db_heap_top - Maximum Database Heap Allocated" on page 607
"db_location - Database Location" on page 607
"db_name - Database Name" on page 608
"db_path - Database Path" on page 609
"db_status - Status of Database" on page 609
"ddl_sql_stmts - Data Definition Language (DDL) SQL Statements" on page 613
"deadlocks - Deadlocks detected monitor element" on page 614
"direct_read_reqs - Direct read requests monitor element" on page 620
"direct_read_time - Direct read time monitor element" on page 622
"direct_reads - Direct reads from database monitor element" on page 624
"direct_write_reqs - Direct write requests monitor element" on page 626
"direct_write_time - Direct write time monitor element" on page 627
"direct_writes - Direct writes to database monitor element" on page 629
"dynamic_sql_stmts - Dynamic SQL Statements Attempted" on page 632
"failed_sql_stmts - Failed Statement Operations" on page 640
"files_closed - Database files closed monitor element" on page 662
"hash_join_overflows - Hash Join Overflows" on page 678
"hash_join_small_overflows - Hash Join Small Overflows" on page 679
"input_db_alias - Input Database Alias" on page 695
"int_auto_rebinds - Internal Automatic Rebinds" on page 697
"int_commits - Internal commits monitor element" on page 698
"int_deadlock_rollbacks - Internal Rollbacks Due To Deadlock" on page 699
"int_rollbacks - Internal rollbacks monitor element" on page 700
"int_rows_deleted - Internal Rows Deleted" on page 702
"int_rows_inserted - Internal Rows Inserted" on page 702

"int_rows_updated - Internal Rows Updated" on page 703
"last_backup - Last Backup Timestamp" on page 711
"last_reset - Last Reset Timestamp" on page 714
"lock_escals - Number of lock escalations monitor element" on page 721
"lock_list_in_use - Total lock list memory in use monitor element" on page 723
"lock_timeouts - Number of lock timeouts monitor element" on page 732
"lock_wait_time - Time waited on locks monitor element" on page 734
"lock_waits - Lock waits monitor element" on page 737
"locks_held - Locks held monitor element" on page 738
"locks_waiting - Current agents waiting on locks monitor element" on page 740
"log_held_by_dirty_pages - Amount of Log Space Accounted for by Dirty Pages" on page 744
"log_read_time - Log Read Time" on page 744
"log_reads - Number of Log Pages Read" on page 745
"log_to_redo_for_recovery - Amount of Log to be Redone for Recovery" on page 745
"log_write_time - Log Write Time" on page 746
"log_writes - Number of Log Pages Written" on page 746
"num_assoc_agents - Number of Associated Agents" on page 771
"num_db_storage_paths - Number of automatic storage paths" on page 773
"num_indoubt_trans - Number of Indoubt Transactions" on page 775
"num_log_buffer_full - Number of times full log buffer caused agents to wait monitor element" on page 775
"num_log_data_found_in_buffer - Number of Log Data Found In Buffer" on page 777
"num_log_part_page_io - Number of Partial Log Page Writes" on page 777
"num_log_read_io - Number of Log Reads" on page 778
"num_log_write_io - Number of Log Writes" on page 778
"olap_func_overflows - OLAP Function Overflows monitor element" on page 782
"pkg_cache_inserts - Package cache inserts monitor element" on page 801
"pkg_cache_lookups - Package cache lookups monitor element" on page 802
"pkg_cache_num_overflows - Package Cache Overflows" on page 804
"pkg_cache_size_top - Package cache high watermark" on page 805
"pool_async_data_read_reqs - Buffer pool asynchronous read requests monitor element" on page 806
"pool_async_data_reads - Buffer pool asynchronous data reads monitor element" on page 807
"pool_async_data_writes - Buffer pool asynchronous data writes monitor element" on page 808
"pool_async_index_read_reqs - Buffer pool asynchronous index read requests monitor element" on page 808
"pool_async_index_reads - Buffer pool asynchronous index reads monitor element" on page 809
"pool_async_index_writes - Buffer pool asynchronous index writes monitor element" on page 810
"pool_async_read_time - Buffer Pool Asynchronous Read Time" on page 811

"pool_async_write_time - Buffer pool asynchronous write time monitor element" on page 812
"pool_async_xda_read_reqs - Buffer pool asynchronous XDA read requests monitor element" on page 812
"pool_async_xda_reads - Buffer pool asynchronous XDA data reads monitor element" on page 813
"pool_async_xda_writes - Buffer pool asynchronous XDA data writes monitor element" on page 814
"pool_data_l_reads - Buffer pool data logical reads monitor element" on page 816
"pool_data_p_reads - Buffer pool data physical reads monitor element" on page 818
"pool_data_writes - Buffer pool data writes monitor element" on page 820
"pool_drty_pg_steal_clns - Buffer pool victim page cleaners triggered monitor element" on page 822
"pool_drty_pg_thrsh_clns - Buffer pool threshold cleaners triggered monitor element" on page 823
"pool_index_l_reads - Buffer pool index logical reads monitor element" on page 825
"pool_index_p_reads - Buffer pool index physical reads monitor element" on page 827
"pool_index_writes - Buffer pool index writes monitor element" on page 829
"pool_lsn_gap_clns - Buffer pool log space cleaners triggered monitor element" on page 831
"pool_no_victim_buffer - Buffer pool no victim buffers monitor element" on page 832
"pool_read_time - Total buffer pool physical read time monitor element" on page 833
"pool_temp_data_l_reads - Buffer pool temporary data logical reads monitor element" on page 835
"pool_temp_data_p_reads - Buffer pool temporary data physical reads monitor element" on page 837
"pool_temp_index_l_reads - Buffer pool temporary index logical reads monitor element" on page 839
"pool_temp_index_p_reads - Buffer pool temporary index physical reads monitor element" on page 841
"pool_temp_xda_l_reads - Buffer pool temporary XDA data logical reads monitor element" on page 843
"pool_temp_xda_p_reads - Buffer pool temporary XDA data physical reads monitor element" on page 845
"pool_write_time - Total buffer pool physical write time monitor element" on page 847
"pool_xda_l_reads - Buffer pool XDA data logical reads monitor element" on page 849
"pool_xda_p_reads - Buffer pool XDA data physical reads monitor element" on page 851
"pool_xda_writes - Buffer pool XDA data writes monitor element" on page 853
"post_shrthreshold_hash_joins - Post threshold hash joins" on page 855

"post_shrthreshold_sorts - Post shared threshold sorts monitor element" on page 856
"priv_workspace_num_overflows - Private Workspace Overflows" on page 862
"priv_workspace_section_inserts - Private Workspace Section Inserts" on page 863
"priv_workspace_section_lookups - Private Workspace Section Lookups" on page 863
"priv_workspace_size_top - Maximum Private Workspace Size" on page 864
"rollback_sql_stmts - Rollback Statements Attempted" on page 886
"rows_deleted - Rows deleted monitor element" on page 889
"rows_inserted - Rows inserted monitor element" on page 890
"rows_read - Rows read monitor element" on page 892
"rows_selected - Rows Selected" on page 895
"rows_updated - Rows updated monitor element" on page 896
"sec_log_used_top - Maximum Secondary Log Space Used" on page 899
"sec_logs_allocated - Secondary Logs Allocated Currently" on page 900
"select_sql_stmts - Select SQL Statements Executed" on page 902
"server_platform - Server Operating System" on page 906
"shr_workspace_num_overflows - Shared Workspace Overflows" on page 911
"shr_workspace_section_inserts - Shared Workspace Section Inserts" on page 911
"shr_workspace_section_lookups - Shared Workspace Section Lookups" on page 912
"shr_workspace_size_top - Maximum Shared Workspace Size" on page 912
"sort_heap_allocated - Total Sort Heap Allocated" on page 914
"sort_overflows - Sort overflows monitor element" on page 915
"sort_shrheap_allocated - Sort Share Heap Currently Allocated" on page 916
"sort_shrheap_top - Sort share heap high watermark" on page 917
"static_sql_stmts - Static SQL Statements Attempted" on page 927
"stats_cache_size – Size of statistics cache monitor element" on page 928
"stats_fabricate_time – Total time spent on statistics fabrication activities monitor element" on page 929
"stats_fabrications – Total number of statistics fabrications monitor elements" on page 929
"sync_runstats – Total number of synchronous RUNSTATS activities monitor element" on page 948
"sync_runstats_time – Total time spent on synchronous RUNSTATS activities monitor element" on page 949
"tot_log_used_top - Maximum Total Log Space Used" on page 987
"total_cons - Connects Since Database Activation" on page 1000
"total_hash_joins - Total Hash Joins" on page 1002
"total_hash_loops - Total Hash Loops" on page 1003
"total_log_available - Total Log Available" on page 1009
"total_log_used - Total Log Space Used" on page 1010
"total_olap_funcs - Total OLAP Functions monitor element" on page 1010
"total_sec_cons - Secondary Connections" on page 1027
"total_sort_time - Total sort time monitor element" on page 1035

"total_sorts - Total sorts monitor element" on page 1036
"uid_sql_stmts - Update/Insert/Delete SQL Statements Executed" on page 1046
"unread_prefetch_pages - Unread prefetch pages monitor element" on page 1047
"x_lock_escals - Exclusive lock escalations monitor element" on page 1066
"xquery_stmts - XQuery Statements Attempted" on page 1068

dbase_remote logical data group

"commit_sql_stmts - Commit Statements Attempted" on page 565
"create_nickname - Create Nicknames" on page 600
"create_nickname_time - Create Nickname Response Time" on page 601
"datasource_name - Data Source Name" on page 605
"db_name - Database Name" on page 608
"delete_sql_stmts - Deletes" on page 617
"delete_time - Delete Response Time" on page 617
"disconnects - Disconnects" on page 631
"failed_sql_stmts - Failed Statement Operations" on page 640
"insert_sql_stmts - Inserts" on page 695
"insert_time - Insert Response Time" on page 696
"passthru_time - Pass-Through Time" on page 799
"passthrus - Pass-Through" on page 800
"remote_lock_time - Remote Lock Time" on page 876
"remote_locks - Remote Locks" on page 877
"rollback_sql_stmts - Rollback Statements Attempted" on page 886
"rows_deleted - Rows deleted monitor element" on page 889
"rows_inserted - Rows inserted monitor element" on page 890
"rows_selected - Rows Selected" on page 895
"rows_updated - Rows updated monitor element" on page 896
"select_sql_stmts - Select SQL Statements Executed" on page 902
"select_time - Query Response Time" on page 903
"sp_rows_selected - Rows Returned by Stored Procedures" on page 918
"stored_proc_time - Stored Procedure Time" on page 947
"stored_procs - Stored Procedures" on page 947
"total_cons - Connects Since Database Activation" on page 1000
"update_sql_stmts - Updates" on page 1052
"update_time - Update Response Time" on page 1053

db_storage_group logical data group

"db_storage_path - Automatic storage path monitor element" on page 610
"fs_id - Unique file system identification number monitor element" on page 664
"fs_total_size - Total size of a file system monitor element" on page 665
"fs_used_size - Amount of space used on a file system monitor element" on page 665
"node_number - Node Number" on page 770
"sto_path_free_sz - Automatic storage path free space monitor element" on page 946

dcs_appl logical data group

"appl_idle_time - Application Idle Time" on page 522

"commit_sql_stmts - Commit Statements Attempted" on page 565

"elapsed_exec_time - Statement Execution Elapsed Time" on page 634

"failed_sql_stmts - Failed Statement Operations" on page 640

"gw_con_time - DB2 Connect Gateway First Connect Initiated" on page 666

"gw_exec_time - Elapsed Time Spent on DB2 Connect Gateway Processing" on page 668

"host_response_time - Host Response Time" on page 682

"inbound_bytes_received - Inbound Number of Bytes Received" on page 692

"inbound_bytes_sent - Inbound Number of Bytes Sent" on page 692

"last_reset - Last Reset Timestamp" on page 714

"max_data_received_1024 - Number of Statements with Outbound Bytes Received Between 513 and 1024 Bytes" on page 748

"max_data_received_128 - Number of Statements with Outbound Bytes Received Between 1 and 128 Bytes" on page 749

"max_data_received_16384 - Number of Statements with Outbound Bytes Received Between 8193 and 16384 Bytes" on page 749

"max_data_received_2048 - Number of Statements with Outbound Bytes Received Between 1025 and 2048 Bytes" on page 750

"max_data_received_256 - Number of Statements with Outbound Bytes Received Between 129 and 256 Bytes" on page 750

"max_data_received_31999 - Number of Statements with Outbound Bytes Received Between 16385 and 31999 Bytes monitor element" on page 751

"max_data_received_4096 - Number of Statements with Outbound Bytes Received Between 2049 and 4096 Bytes" on page 751

"max_data_received_512 - Number of Statements with Outbound Bytes Received Between 257 and 512 Bytes" on page 751

"max_data_received_64000 - Number of Statements with Outbound Bytes Received Between 32000 and 64000 Bytes monitor element" on page 752

"max_data_received_8192 - Number of Statements with Outbound Bytes Received Between 4097 and 8192 Bytes" on page 752

"max_data_received_gt64000 - Number of Statements with Outbound Bytes Received Greater than 64000 Bytes" on page 753

"max_data_sent_1024 - Number of Statements with Outbound Bytes Sent Between 513 and 1024 Bytes" on page 753

"max_data_sent_128 - Number of Statements with Outbound Bytes Sent Between 1 and 128 Bytes" on page 753

"max_data_sent_16384 - Number of Statements with Outbound Bytes Sent Between 8193 and 16384 Bytes" on page 754

"max_data_sent_2048 - Number of Statements with Outbound Bytes Sent Between 1025 and 2048 Bytes" on page 754

"max_data_sent_256 - Number of Statements with Outbound Bytes Sent Between 129 and 256 Bytes" on page 755

"max_data_sent_31999 - Number of Statements with Outbound Bytes Sent Between 16385 and 31999 Bytes" on page 755

"max_data_sent_4096 - Number of Statements with Outbound Bytes Sent Between 2049 and 4096 Bytes" on page 756

"max_data_sent_512 - Number of Statements with Outbound Bytes Sent Between 257 and 512 Bytes" on page 756
"max_data_sent_64000 - Number of Statements with Outbound Bytes Sent Between 32000 and 64000 Bytes" on page 756
"max_data_sent_8192 - Number of Statements with Outbound Bytes Sent Between 4097 and 8192 Bytes" on page 757
"max_data_sent_gt64000 - Number of Statements with Outbound Bytes Sent Greater than 64000 Bytes" on page 757
"max_network_time_100_ms - Number of Statements with Network Time between 16 and 100 ms" on page 758
"max_network_time_16_ms - Number of Statements with Network Time between 4 and 16 ms" on page 758
"max_network_time_1_ms - Number of Statements with Network Time of up to 1 ms" on page 759
"max_network_time_4_ms - Number of Statements with Network Time between 1 and 4 ms" on page 759
"max_network_time_500_ms - Number of Statements with Network Time between 100 and 500 ms" on page 759
"max_network_time_gt500_ms - Number of Statements with Network Time greater than 500 ms" on page 760
"network_time_bottom - Minimum Network Time for Statement" on page 768
"network_time_top - Maximum Network Time for Statement" on page 769
"open_cursors - Number of Open Cursors" on page 782
"outbound_bytes_received - Outbound Number of Bytes Received" on page 786
"outbound_bytes_sent - Outbound Number of Bytes Sent" on page 787
"prev_uow_stop_time - Previous Unit of Work Completion Timestamp" on page 861
"rollback_sql_stmts - Rollback Statements Attempted" on page 886
"rows_selected - Rows Selected" on page 895
"sql_stmts - Number of SQL Statements Attempted" on page 919
"tpmon_acc_str - TP monitor client accounting string monitor element" on page 1039
"tpmon_client_app - TP monitor client application name monitor element" on page 1040
"tpmon_client_userid - TP monitor client user ID monitor element" on page 1040
"tpmon_client_wkstn - TP monitor client workstation name monitor element" on page 1041
"uow_comp_status - Unit of Work Completion Status" on page 1047
"uow_elapsed_time - Most Recent Unit of Work Elapsed Time" on page 1048
"uow_start_time - Unit of work start timestamp monitor element" on page 1050
"uow_stop_time - Unit of work stop timestamp monitor element" on page 1051
"xid - Transaction ID" on page 1068

dcs_appl_info logical data group

"agent_id - Application handle (agent ID) monitor element" on page 506
"agent_status - DCS Application Agents" on page 509
"appl_id - Application ID monitor element" on page 519

"appl_name - Application name monitor element" on page 523
"auth_id - Authorization ID" on page 537
"client_pid - Client process ID monitor element" on page 558
"client_platform - Client operating platform monitor element" on page 559
"client_prdid - Client product and version ID monitor element" on page 560
"client_protocol - Client communication protocol monitor element" on page 561
"codepage_id - ID of Code Page Used by Application" on page 564
"dcs_appl_status - DCS Application Status" on page 612
"dcs_db_name - DCS Database Name" on page 612
"execution_id - User Login ID" on page 639
"gw_db_alias - Database Alias at the Gateway" on page 668
"host_ccsid - Host Coded Character Set ID" on page 681
"host_db_name - Host Database Name" on page 681
"host_prdid - Host Product/Version ID" on page 681
"inbound_comm_address - Inbound Communication Address" on page 693
"outbound_appl_id - Outbound Application ID" on page 785
"outbound_comm_address - Outbound Communication Address" on page 788
"outbound_comm_protocol - Outbound Communication Protocol" on page 789
"outbound_sequence_no - Outbound Sequence Number" on page 789
"sequence_no - Sequence number monitor element" on page 904
"status_change_time - Application Status Change Time" on page 930

dcs_dbase logical data group

"commit_sql_stmts - Commit Statements Attempted" on page 565
"con_elapsed_time - Most Recent Connection Elapsed Time" on page 567
"con_response_time - Most Recent Response Time for Connect" on page 567
"dcs_db_name - DCS Database Name" on page 612
"elapsed_exec_time - Statement Execution Elapsed Time" on page 634
"failed_sql_stmts - Failed Statement Operations" on page 640
"gw_comm_error_time - Communication Error Time" on page 666
"gw_comm_errors - Communication Errors" on page 666
"gw_con_time - DB2 Connect Gateway First Connect Initiated" on page 666
"gw_connections_top - Maximum Number of Concurrent Connections to Host Database" on page 666
"gw_cons_wait_client - Number of Connections Waiting for the Client to Send Request" on page 667
"gw_cons_wait_host - Number of Connections Waiting for the Host to Reply" on page 667
"gw_cur_cons - Current Number of Connections for DB2 Connect" on page 668
"gw_total_cons - Total Number of Attempted Connections for DB2 Connect" on page 669
"host_db_name - Host Database Name" on page 681
"host_response_time - Host Response Time" on page 682
"inbound_bytes_received - Inbound Number of Bytes Received" on page 692
"last_reset - Last Reset Timestamp" on page 714

"max_data_received_1024 - Number of Statements with Outbound Bytes Received Between 513 and 1024 Bytes" on page 748

"max_data_received_128 - Number of Statements with Outbound Bytes Received Between 1 and 128 Bytes" on page 749

"max_data_received_16384 - Number of Statements with Outbound Bytes Received Between 8193 and 16384 Bytes" on page 749

"max_data_received_2048 - Number of Statements with Outbound Bytes Received Between 1025 and 2048 Bytes" on page 750

"max_data_received_256 - Number of Statements with Outbound Bytes Received Between 129 and 256 Bytes" on page 750

"max_data_received_31999 - Number of Statements with Outbound Bytes Received Between 16385 and 31999 Bytes monitor element" on page 751

"max_data_received_4096 - Number of Statements with Outbound Bytes Received Between 2049 and 4096 Bytes" on page 751

"max_data_received_512 - Number of Statements with Outbound Bytes Received Between 257 and 512 Bytes" on page 751

"max_data_received_64000 - Number of Statements with Outbound Bytes Received Between 32000 and 64000 Bytes monitor element" on page 752

"max_data_received_8192 - Number of Statements with Outbound Bytes Received Between 4097 and 8192 Bytes" on page 752

"max_data_received_gt64000 - Number of Statements with Outbound Bytes Received Greater than 64000 Bytes" on page 753

"max_data_sent_1024 - Number of Statements with Outbound Bytes Sent Between 513 and 1024 Bytes" on page 753

"max_data_sent_128 - Number of Statements with Outbound Bytes Sent Between 1 and 128 Bytes" on page 753

"max_data_sent_16384 - Number of Statements with Outbound Bytes Sent Between 8193 and 16384 Bytes" on page 754

"max_data_sent_2048 - Number of Statements with Outbound Bytes Sent Between 1025 and 2048 Bytes" on page 754

"max_data_sent_256 - Number of Statements with Outbound Bytes Sent Between 129 and 256 Bytes" on page 755

"max_data_sent_31999 - Number of Statements with Outbound Bytes Sent Between 16385 and 31999 Bytes" on page 755

"max_data_sent_4096 - Number of Statements with Outbound Bytes Sent Between 2049 and 4096 Bytes" on page 756

"max_data_sent_512 - Number of Statements with Outbound Bytes Sent Between 257 and 512 Bytes" on page 756

"max_data_sent_64000 - Number of Statements with Outbound Bytes Sent Between 32000 and 64000 Bytes" on page 756

"max_data_sent_8192 - Number of Statements with Outbound Bytes Sent Between 4097 and 8192 Bytes" on page 757

"max_data_sent_gt64000 - Number of Statements with Outbound Bytes Sent Greater than 64000 Bytes" on page 757

"max_network_time_100_ms - Number of Statements with Network Time between 16 and 100 ms" on page 758

"max_network_time_16_ms - Number of Statements with Network Time between 4 and 16 ms" on page 758

"max_network_time_1_ms - Number of Statements with Network Time of up to 1 ms" on page 759

"max_network_time_4_ms - Number of Statements with Network Time between 1 and 4 ms" on page 759
"max_network_time_500_ms - Number of Statements with Network Time between 100 and 500 ms" on page 759
"max_network_time_gt500_ms - Number of Statements with Network Time greater than 500 ms" on page 760
"network_time_bottom - Minimum Network Time for Statement" on page 768
"network_time_top - Maximum Network Time for Statement" on page 769
"outbound_bytes_sent - Outbound Number of Bytes Sent" on page 787
"rollback_sql_stmts - Rollback Statements Attempted" on page 886
"rows_selected - Rows Selected" on page 895
"sql_stmts - Number of SQL Statements Attempted" on page 919

dcs_stmt logical data group

"blocking_cursor - Blocking Cursor" on page 542
"creator - Application Creator" on page 601
"elapsed_exec_time - Statement Execution Elapsed Time" on page 634
"fetch_count - Number of Successful Fetches" on page 661
"gw_exec_time - Elapsed Time Spent on DB2 Connect Gateway Processing" on page 668
"host_response_time - Host Response Time" on page 682
"inbound_bytes_received - Inbound Number of Bytes Received" on page 692
"inbound_bytes_sent - Inbound Number of Bytes Sent" on page 692
"num_transmissions - Number of Transmissions" on page 781
"num_transmissions_group - Number of Transmissions Group" on page 781
"outbound_bytes_received - Outbound Number of Bytes Received" on page 786
"outbound_bytes_sent - Outbound Number of Bytes Sent" on page 787
"package_name - Package name monitor element" on page 791
"query_card_estimate - Query Number of Rows Estimate" on page 868
"query_cost_estimate - Query cost estimate monitor element" on page 869
"section_number - Section number monitor element" on page 901
"stmt_elapsed_time - Most Recent Statement Elapsed Time" on page 930
"stmt_operation/operation - Statement operation monitor element" on page 935
"stmt_start - Statement Operation Start Timestamp" on page 939
"stmt_stop - Statement Operation Stop Timestamp" on page 939
"stmt_text - SQL statement text monitor element" on page 940

detail_log logical data group

"current_active_log - Current Active Log File Number" on page 602
"current_archive_log - Current Archive Log File Number" on page 602
"first_active_log - First Active Log File Number" on page 663
"last_active_log - Last Active Log File Number" on page 711
"node_number - Node Number" on page 770

dynsql logical data group

"fetch_count - Number of Successful Fetches" on page 661
"insert_timestamp - Insert timestamp monitor element" on page 696

"int_rows_deleted - Internal Rows Deleted" on page 702
"int_rows_inserted - Internal Rows Inserted" on page 702
"int_rows_updated - Internal Rows Updated" on page 703
"num_compilations - Statement Compilations" on page 772
"num_executions - Statement executions monitor element" on page 773
"pool_data_l_reads - Buffer pool data logical reads monitor element" on page 816
"pool_data_p_reads - Buffer pool data physical reads monitor element" on page 818
"pool_index_l_reads - Buffer pool index logical reads monitor element" on page 825
"pool_index_p_reads - Buffer pool index physical reads monitor element" on page 827
"pool_temp_data_l_reads - Buffer pool temporary data logical reads monitor element" on page 835
"pool_temp_data_p_reads - Buffer pool temporary data physical reads monitor element" on page 837
"pool_temp_index_l_reads - Buffer pool temporary index logical reads monitor element" on page 839
"pool_temp_index_p_reads - Buffer pool temporary index physical reads monitor element" on page 841
"pool_temp_xda_l_reads - Buffer pool temporary XDA data logical reads monitor element" on page 843
"pool_temp_xda_p_reads - Buffer pool temporary XDA data physical reads monitor element" on page 845
"pool_xda_l_reads - Buffer pool XDA data logical reads monitor element" on page 849
"pool_xda_p_reads - Buffer pool XDA data physical reads monitor element" on page 851
"prep_time_best - Statement best preparation time monitor element" on page 861
"prep_time_worst - Statement worst preparation time monitor element" on page 861
"rows_read - Rows read monitor element" on page 892
"rows_written - Rows Written" on page 897
"sort_overflows - Sort overflows monitor element" on page 915
"stats_fabricate_time – Total time spent on statistics fabrication activities monitor element" on page 929
"stmt_pkgcache_id - Statement package cache identifier monitor element" on page 937
"stmt_sorts - Statement Sorts" on page 938
"stmt_text - SQL statement text monitor element" on page 940
"sync_runstats_time – Total time spent on synchronous RUNSTATS activities monitor element" on page 949
"total_exec_time - Elapsed statement execution time monitor element" on page 1002
"total_sort_time - Total sort time monitor element" on page 1035
"total_sys_cpu_time - Total system CPU time for a statement monitor element" on page 1037

"total_usr_cpu_time - Total user CPU time for a statement monitor element" on page 1038

dynsql_list logical data group

"db_name - Database Name" on page 608
"db_path - Database Path" on page 609

fcm logical data group

"buff_free - FCM buffers currently free monitor element" on page 545
"buff_free_bottom - Minimum FCM buffers free monitor element" on page 545
"buff_max - Maximum possible number of FCM buffers monitor element" on page 546
"buff_total - Number of currently allocated FCM buffers monitor element" on page 546
"ch_free - Channels currently free monitor element" on page 552
"ch_free_bottom - Minimum channels free monitor element" on page 553
"ch_max - Maximum possible number of FCM channels monitor element" on page 553
"ch_total - Number of currently allocated FCM channels monitor element" on page 554

fcm_node logical data group

"connection_status - Connection status monitor element" on page 579
"node_number - Node Number" on page 770
"total_buffers_rcvd - Total FCM buffers received monitor element" on page 994
"total_buffers_sent - Total FCM buffers sent monitor element" on page 994

hadr logical data group

"hadr_connect_status - HADR Connection Status monitor element" on page 669
"hadr_connect_time - HADR Connection Time monitor element" on page 670
"hadr_heartbeat - HADR Heartbeat monitor element" on page 670
"hadr_local_host - HADR Local Host monitor element" on page 671
"hadr_local_service - HADR Local Service monitor element" on page 671
"hadr_log_gap - HADR Log Gap" on page 672
"hadr_primary_log_file - HADR Primary Log File monitor element" on page 673
"hadr_primary_log_lsn - HADR Primary Log LSN monitor element" on page 673
"hadr_primary_log_page - HADR Primary Log Page monitor element" on page 674
"hadr_remote_host - HADR Remote Host monitor element" on page 674
"hadr_remote_instance - HADR Remote Instance monitor element" on page 674
"hadr_remote_service - HADR Remote Service monitor element" on page 675
"hadr_role - HADR Role" on page 675
"hadr_standby_log_file - HADR Standby Log File monitor element" on page 676
"hadr_standby_log_lsn - HADR Standby Log LSN monitor element" on page 676

"hadr_standby_log_page - HADR Standby Log Page monitor element" on page 676
"hadr_state - HADR State monitor element" on page 677
"hadr_syncmode - HADR Synchronization Mode monitor element" on page 677
"hadr_timeout - HADR Timeout monitor element" on page 678

lock logical data group

"data_partition_id - Data partition identifier monitor element" on page 604
"lock_attributes - Lock attributes monitor element" on page 717
"lock_count - Lock count monitor element" on page 718
"lock_current_mode - Original lock mode before conversion monitor element" on page 719
"lock_escalation - Lock escalation monitor element" on page 720
"lock_hold_count - Lock hold count monitor element" on page 723
"lock_mode - Lock mode monitor element" on page 724
"lock_name - Lock name monitor element" on page 726
"lock_object_name - Lock Object Name" on page 727
"lock_object_type - Lock object type waited on monitor element" on page 728
"lock_release_flags - Lock release flags monitor element" on page 730
"lock_status - Lock status monitor element" on page 730
"node_number - Node Number" on page 770
"table_file_id - Table file ID monitor element" on page 951
"table_name - Table name monitor element" on page 952
"table_schema - Table schema name monitor element" on page 953
"tablespace_name - Table space name monitor element" on page 961

lock_wait logical data group

"agent_id_holding_lock - Agent ID Holding Lock" on page 508
"appl_id_holding_lk - Application ID Holding Lock" on page 521
"data_partition_id - Data partition identifier monitor element" on page 604
"lock_attributes - Lock attributes monitor element" on page 717
"lock_current_mode - Original lock mode before conversion monitor element" on page 719
"lock_escalation - Lock escalation monitor element" on page 720
"lock_mode - Lock mode monitor element" on page 724
"lock_mode_requested - Lock mode requested monitor element" on page 725
"lock_name - Lock name monitor element" on page 726
"lock_object_type - Lock object type waited on monitor element" on page 728
"lock_release_flags - Lock release flags monitor element" on page 730
"lock_wait_start_time - Lock wait start timestamp monitor element" on page 734
"node_number - Node Number" on page 770
"ss_number - Subsection number monitor element" on page 925
"table_name - Table name monitor element" on page 952
"table_schema - Table schema name monitor element" on page 953
"tablespace_name - Table space name monitor element" on page 961

memory_pool logical data group

"node_number - Node Number" on page 770
"pool_config_size - Configured Size of Memory Pool" on page 815
"pool_cur_size - Current Size of Memory Pool" on page 815
"pool_id - Memory Pool Identifier" on page 824
"pool_secondary_id - Memory Pool Secondary Identifier" on page 834
"pool_watermark - Memory Pool Watermark" on page 847

progress logical data group

"progress_completed_units - Completed Progress Work Units" on page 865
"progress_description - Progress Description" on page 865
"progress_seq_num - Progress Sequence Number" on page 866
"progress_start_time - Progress Start Time" on page 867
"progress_total_units - Total Progress Work Units" on page 867
"progress_work_metric - Progress Work Metric" on page 867

progress_list logical data group

"progress_list_attr - Current Progress List Attributes" on page 866
"progress_list_cur_seq_num - Current Progress List Sequence Number" on page 866

rollforward logical data group

"node_number - Node Number" on page 770
"rf_log_num - Log Being Rolled Forward" on page 884
"rf_status - Log Phase" on page 885
"rf_timestamp - Rollforward Timestamp" on page 885
"rf_type - Rollforward Type" on page 885
"ts_name - Table space being rolled forward monitor element" on page 1046

stmt logical data group

"agents_top - Number of Agents Created" on page 515
"blocking_cursor - Blocking Cursor" on page 542
"consistency_token - Package consistency token monitor element" on page 580
"creator - Application Creator" on page 601
"cursor_name - Cursor Name" on page 603
"degree_parallelism - Degree of Parallelism" on page 616
"fetch_count - Number of Successful Fetches" on page 661
"int_rows_deleted - Internal Rows Deleted" on page 702
"int_rows_inserted - Internal Rows Inserted" on page 702
"int_rows_updated - Internal Rows Updated" on page 703
"num_agents - Number of Agents Working on a Statement" on page 771
"package_name - Package name monitor element" on page 791
"package_version_id - Package version monitor element" on page 792
"pool_data_l_reads - Buffer pool data logical reads monitor element" on page 816
"pool_data_p_reads - Buffer pool data physical reads monitor element" on page 818

"pool_index_l_reads - Buffer pool index logical reads monitor element" on page 825
"pool_index_p_reads - Buffer pool index physical reads monitor element" on page 827
"pool_temp_data_l_reads - Buffer pool temporary data logical reads monitor element" on page 835
"pool_temp_data_p_reads - Buffer pool temporary data physical reads monitor element" on page 837
"pool_temp_index_l_reads - Buffer pool temporary index logical reads monitor element" on page 839
"pool_temp_index_p_reads - Buffer pool temporary index physical reads monitor element" on page 841
"pool_temp_xda_l_reads - Buffer pool temporary XDA data logical reads monitor element" on page 843
"pool_temp_xda_p_reads - Buffer pool temporary XDA data physical reads monitor element" on page 845
"pool_xda_l_reads - Buffer pool XDA data logical reads monitor element" on page 849
"pool_xda_p_reads - Buffer pool XDA data physical reads monitor element" on page 851
"query_card_estimate - Query Number of Rows Estimate" on page 868
"query_cost_estimate - Query cost estimate monitor element" on page 869
"rows_read - Rows read monitor element" on page 892
"rows_written - Rows Written" on page 897
"section_number - Section number monitor element" on page 901
"sort_overflows - Sort overflows monitor element" on page 915
"stmt_elapsed_time - Most Recent Statement Elapsed Time" on page 930
"stmt_node_number - Statement Node" on page 935
"stmt_operation/operation - Statement operation monitor element" on page 935
"stmt_sorts - Statement Sorts" on page 938
"stmt_start - Statement Operation Start Timestamp" on page 939
"stmt_stop - Statement Operation Stop Timestamp" on page 939
"stmt_sys_cpu_time - System CPU Time used by Statement" on page 940
"stmt_text - SQL statement text monitor element" on page 940
"stmt_type - Statement type monitor element" on page 942
"stmt_usr_cpu_time - User CPU Time used by Statement" on page 943
"total_sort_time - Total sort time monitor element" on page 1035

stmt_transmissions logical data group

"elapsed_exec_time - Statement Execution Elapsed Time" on page 634
"host_response_time - Host Response Time" on page 682
"max_data_received_1024 - Number of Statements with Outbound Bytes Received Between 513 and 1024 Bytes" on page 748
"max_data_received_128 - Number of Statements with Outbound Bytes Received Between 1 and 128 Bytes" on page 749
"max_data_received_16384 - Number of Statements with Outbound Bytes Received Between 8193 and 16384 Bytes" on page 749

"max_data_received_2048 - Number of Statements with Outbound Bytes Received Between 1025 and 2048 Bytes" on page 750

"max_data_received_256 - Number of Statements with Outbound Bytes Received Between 129 and 256 Bytes" on page 750

"max_data_received_31999 - Number of Statements with Outbound Bytes Received Between 16385 and 31999 Bytes monitor element" on page 751

"max_data_received_4096 - Number of Statements with Outbound Bytes Received Between 2049 and 4096 Bytes" on page 751

"max_data_received_512 - Number of Statements with Outbound Bytes Received Between 257 and 512 Bytes" on page 751

"max_data_received_64000 - Number of Statements with Outbound Bytes Received Between 32000 and 64000 Bytes monitor element" on page 752

"max_data_received_8192 - Number of Statements with Outbound Bytes Received Between 4097 and 8192 Bytes" on page 752

"max_data_received_gt64000 - Number of Statements with Outbound Bytes Received Greater than 64000 Bytes" on page 753

"max_data_sent_1024 - Number of Statements with Outbound Bytes Sent Between 513 and 1024 Bytes" on page 753

"max_data_sent_128 - Number of Statements with Outbound Bytes Sent Between 1 and 128 Bytes" on page 753

"max_data_sent_16384 - Number of Statements with Outbound Bytes Sent Between 8193 and 16384 Bytes" on page 754

"max_data_sent_2048 - Number of Statements with Outbound Bytes Sent Between 1025 and 2048 Bytes" on page 754

"max_data_sent_256 - Number of Statements with Outbound Bytes Sent Between 129 and 256 Bytes" on page 755

"max_data_sent_31999 - Number of Statements with Outbound Bytes Sent Between 16385 and 31999 Bytes" on page 755

"max_data_sent_4096 - Number of Statements with Outbound Bytes Sent Between 2049 and 4096 Bytes" on page 756

"max_data_sent_512 - Number of Statements with Outbound Bytes Sent Between 257 and 512 Bytes" on page 756

"max_data_sent_64000 - Number of Statements with Outbound Bytes Sent Between 32000 and 64000 Bytes" on page 756

"max_data_sent_8192 - Number of Statements with Outbound Bytes Sent Between 4097 and 8192 Bytes" on page 757

"max_data_sent_gt64000 - Number of Statements with Outbound Bytes Sent Greater than 64000 Bytes" on page 757

"max_network_time_100_ms - Number of Statements with Network Time between 16 and 100 ms" on page 758

"max_network_time_16_ms - Number of Statements with Network Time between 4 and 16 ms" on page 758

"max_network_time_1_ms - Number of Statements with Network Time of up to 1 ms" on page 759

"max_network_time_4_ms - Number of Statements with Network Time between 1 and 4 ms" on page 759

"max_network_time_500_ms - Number of Statements with Network Time between 100 and 500 ms" on page 759

"max_network_time_gt500_ms - Number of Statements with Network Time greater than 500 ms" on page 760

"network_time_bottom - Minimum Network Time for Statement" on page 768

"network_time_top - Maximum Network Time for Statement" on page 769

"outbound_bytes_received - Outbound Number of Bytes Received" on page 786

"outbound_bytes_received_bottom - Minimum Outbound Number of Bytes Received" on page 787

"outbound_bytes_received_top - Maximum Outbound Number of Bytes Received" on page 787

"outbound_bytes_sent - Outbound Number of Bytes Sent" on page 787

"outbound_bytes_sent_bottom - Minimum Outbound Number of Bytes Sent" on page 788

"outbound_bytes_sent_top - Maximum Outbound Number of Bytes Sent" on page 788

"sql_chains - Number of SQL Chains Attempted" on page 918

"sql_stmts - Number of SQL Statements Attempted" on page 919

subsection logical data group

"rows_read - Rows read monitor element" on page 892

"rows_written - Rows Written" on page 897

"ss_exec_time - Subsection Execution Elapsed Time" on page 924

"ss_node_number - Subsection Node Number" on page 925

"ss_number - Subsection number monitor element" on page 925

"ss_status - Subsection Status" on page 925

"ss_sys_cpu_time - System CPU Time used by Subsection" on page 926

"ss_usr_cpu_time - User CPU Time used by Subsection" on page 926

"tq_cur_send_spills - Current number of table queue buffers overflowed monitor element" on page 1042

"tq_id_waiting_on - Waited on node on a table queue monitor element" on page 1042

"tq_max_send_spills - Maximum number of table queue buffers overflows" on page 1042

"tq_node_waited_for - Waited for node on a table queue" on page 1043

"tq_rows_read - Number of Rows Read from table queues" on page 1043

"tq_rows_written - Number of rows written to table queues" on page 1044

"tq_tot_send_spills - Total number of table queue buffers overflowed monitor element" on page 1044

"tq_wait_for_any - Waiting for any node to send on a table queue" on page 1045

table logical data group

"data_object_pages - Data Object Pages" on page 603

"data_partition_id - Data partition identifier monitor element" on page 604

"index_object_pages - Index Object Pages" on page 693

"lob_object_pages - LOB Object Pages" on page 715

"long_object_pages - Long Object Pages" on page 747

"overflow_accesses - Accesses to overflowed records monitor element" on page 789

"page_reorgs - Page reorganizations monitor element" on page 794

"rows_read - Rows read monitor element" on page 892

"rows_written - Rows Written" on page 897
"table_file_id - Table file ID monitor element" on page 951
"table_name - Table name monitor element" on page 952
"table_schema - Table schema name monitor element" on page 953
"table_type - Table type monitor element" on page 954
"tablespace_id - Table space identification monitor element" on page 958
"xda_object_pages - XDA Object Pages" on page 1067

table_list logical data group

"db_conn_time - Database activation timestamp monitor element" on page 607
"db_name - Database Name" on page 608
"db_path - Database Path" on page 609
"input_db_alias - Input Database Alias" on page 695
"last_reset - Last Reset Timestamp" on page 714

table_reorg logical data group

"data_partition_id - Data partition identifier monitor element" on page 604
"reorg_completion - Reorganization Completion Flag" on page 877
"reorg_current_counter - Reorganize Progress" on page 878
"reorg_end - Table Reorganize End Time" on page 878
"reorg_index_id - Index Used to Reorganize the Table" on page 878
"reorg_max_counter - Total Amount of Reorganization" on page 879
"reorg_max_phase - Maximum Reorganize Phase" on page 879
"reorg_phase - Table reorganization phase monitor element" on page 879
"reorg_phase_start - Reorganize Phase Start Time" on page 880
"reorg_rows_compressed - Rows Compressed" on page 880
"reorg_rows_rejected_for_compression - Rows Rejected for Compression" on page 880
"reorg_start - Table Reorganize Start Time" on page 881
"reorg_status - Table Reorganize Status" on page 881
"reorg_tbspc_id - Table Space Where Table or Data partition is Reorganized" on page 881
"reorg_type - Table Reorganize Attributes" on page 882
"reorg_xml_regions_compressed – XML regions compressed monitor element" on page 882
"reorg_xml_regions_rejected_for_compression – XML regions rejected for compression monitor element" on page 883

tablespace logical data group

"direct_read_reqs - Direct read requests monitor element" on page 620
"direct_read_time - Direct read time monitor element" on page 622
"direct_reads - Direct reads from database monitor element" on page 624
"direct_write_reqs - Direct write requests monitor element" on page 626
"direct_write_time - Direct write time monitor element" on page 627
"direct_writes - Direct writes to database monitor element" on page 629
"files_closed - Database files closed monitor element" on page 662
"fs_caching - File system caching monitor element" on page 664

"pool_async_data_read_reqs - Buffer pool asynchronous read requests monitor element" on page 806
"pool_async_data_reads - Buffer pool asynchronous data reads monitor element" on page 807
"pool_async_data_writes - Buffer pool asynchronous data writes monitor element" on page 808
"pool_async_index_read_reqs - Buffer pool asynchronous index read requests monitor element" on page 808
"pool_async_index_reads - Buffer pool asynchronous index reads monitor element" on page 809
"pool_async_index_writes - Buffer pool asynchronous index writes monitor element" on page 810
"pool_async_read_time - Buffer Pool Asynchronous Read Time" on page 811
"pool_async_write_time - Buffer pool asynchronous write time monitor element" on page 812
"pool_async_xda_read_reqs - Buffer pool asynchronous XDA read requests monitor element" on page 812
"pool_async_xda_reads - Buffer pool asynchronous XDA data reads monitor element" on page 813
"pool_async_xda_writes - Buffer pool asynchronous XDA data writes monitor element" on page 814
"pool_data_l_reads - Buffer pool data logical reads monitor element" on page 816
"pool_data_p_reads - Buffer pool data physical reads monitor element" on page 818
"pool_data_writes - Buffer pool data writes monitor element" on page 820
"pool_index_l_reads - Buffer pool index logical reads monitor element" on page 825
"pool_index_p_reads - Buffer pool index physical reads monitor element" on page 827
"pool_index_writes - Buffer pool index writes monitor element" on page 829
"pool_no_victim_buffer - Buffer pool no victim buffers monitor element" on page 832
"pool_read_time - Total buffer pool physical read time monitor element" on page 833
"pool_temp_data_l_reads - Buffer pool temporary data logical reads monitor element" on page 835
"pool_temp_data_p_reads - Buffer pool temporary data physical reads monitor element" on page 837
"pool_temp_index_l_reads - Buffer pool temporary index logical reads monitor element" on page 839
"pool_temp_index_p_reads - Buffer pool temporary index physical reads monitor element" on page 841
"pool_temp_xda_l_reads - Buffer pool temporary XDA data logical reads monitor element" on page 843
"pool_temp_xda_p_reads - Buffer pool temporary XDA data physical reads monitor element" on page 845
"pool_write_time - Total buffer pool physical write time monitor element" on page 847

"pool_xda_l_reads - Buffer pool XDA data logical reads monitor element" on page 849
"pool_xda_p_reads - Buffer pool XDA data physical reads monitor element" on page 851
"pool_xda_writes - Buffer pool XDA data writes monitor element" on page 853
"tablespace_auto_resize_enabled - Table space automatic resizing enabled monitor element" on page 955
"tablespace_content_type - Table space content type monitor element" on page 956
"tablespace_cur_pool_id - Buffer pool currently being used monitor element" on page 956
"tablespace_extent_size - Table space extent size monitor element" on page 957
"tablespace_id - Table space identification monitor element" on page 958
"tablespace_name - Table space name monitor element" on page 961
"tablespace_next_pool_id - Buffer pool that will be used at next startup monitor element" on page 962
"tablespace_page_size - Table space page size monitor element" on page 963
"tablespace_prefetch_size - Table space prefetch size monitor element" on page 965
"tablespace_rebalancer_mode - Rebalancer mode monitor element" on page 966
"tablespace_type - Table space type monitor element" on page 973
"tablespace_using_auto_storage - Table space enabled for automatic storage monitor element" on page 974

tablespace_container logical data group

"container_accessible - Accessibility of container monitor element" on page 581
"container_id - Container identification monitor element" on page 581
"container_name - Container name monitor element" on page 581
"container_stripe_set - Container stripe set monitor element" on page 582
"container_total_pages - Total pages in container monitor element" on page 582
"container_type - Container type monitor element" on page 583
"container_usable_pages - Usable pages in container monitor element" on page 583

tablespace_list logical data group

"db_conn_time - Database activation timestamp monitor element" on page 607
"db_name - Database Name" on page 608
"db_path - Database Path" on page 609
"input_db_alias - Input Database Alias" on page 695
"last_reset - Last Reset Timestamp" on page 714

tablespace_nodeinfo logical data group

"tablespace_current_size - Current table space size" on page 956
"tablespace_free_pages - Free pages in table space monitor element" on page 957
"tablespace_increase_size - Increase size in bytes" on page 958
"tablespace_increase_size_percent - Increase size by percent monitor element" on page 959

"tablespace_initial_size - Initial table space size" on page 959
"tablespace_last_resize_failed - Last resize attempt failed" on page 959
"tablespace_last_resize_time - Time of last successful resize" on page 960
"tablespace_max_size - Maximum table space size" on page 960
"tablespace_min_recovery_time - Minimum recovery time for rollforward monitor element" on page 960
"tablespace_num_containers - Number of Containers in Table Space" on page 962
"tablespace_num_quiescers - Number of Quiescers" on page 962
"tablespace_num_ranges - Number of Ranges in the Table Space Map" on page 963
"tablespace_page_top - Table space high watermark monitor element" on page 963
"tablespace_paths_dropped - Table space using dropped path monitor element" on page 964
"tablespace_pending_free_pages - Pending free pages in table space monitor element" on page 964
"tablespace_prefetch_size - Table space prefetch size monitor element" on page 965
"tablespace_rebalancer_extents_processed - Number of Extents the Rebalancer has Processed" on page 965
"tablespace_rebalancer_extents_remaining - Total Number of Extents to be Processed by the Rebalancer" on page 965
"tablespace_rebalancer_last_extent_moved - Last Extent Moved by the Rebalancer" on page 966
"tablespace_rebalancer_priority - Current Rebalancer Priority" on page 968
"tablespace_rebalancer_restart_time - Rebalancer Restart Time" on page 968
"tablespace_rebalancer_start_time - Rebalancer Start Time" on page 968
"tablespace_state - Table space state monitor element" on page 969
"tablespace_state_change_object_id - State Change Object Identification" on page 971
"tablespace_state_change_ts_id - State Change Table Space Identification" on page 971
"tablespace_total_pages - Total pages in table space monitor element" on page 971
"tablespace_usable_pages - Usable pages in table space monitor element" on page 973
"tablespace_used_pages - Used pages in table space monitor element" on page 974

tablespace_quiescer logical data group

"quiescer_agent_id - Quiescer Agent Identification" on page 871
"quiescer_auth_id - Quiescer User Authorization Identification" on page 871
"quiescer_obj_id - Quiescer Object Identification" on page 871
"quiescer_state - Quiescer State" on page 872
"quiescer_ts_id - Quiescer Table Space Identification" on page 872

tablespace_range logical data group

"range_adjustment - Range Adjustment" on page 872
"range_container_id - Range Container" on page 873
"range_end_stripe - End Stripe" on page 873
"range_max_extent - Maximum Extent in Range" on page 873
"range_max_page_number - Maximum Page in Range" on page 873
"range_num_containers - Number of Containers in Range" on page 873
"range_number - Range Number" on page 874
"range_offset - Range Offset" on page 874
"range_start_stripe - Start Stripe" on page 874
"range_stripe_set_number - Stripe Set Number" on page 874

utility_info logical data group

"node_number - Node Number" on page 770
"utility_dbname - Database Operated on by Utility" on page 1054
"utility_description - Utility Description" on page 1054
"utility_id - Utility ID" on page 1054
"utility_invoker_type - Utility Invoker Type" on page 1055
"utility_priority - Utility Priority" on page 1055
"utility_start_time - Utility Start Time" on page 1055
"utility_state - Utility State" on page 1055
"utility_type - Utility Type" on page 1056

Event type mappings to logical data groups

For file, pipe and table event monitors, event monitor output consists of an ordered series of logical data groupings. Regardless of the event monitor type, the output records always contain the same starting logical data groups. These frame the logical data groups whose presence varies depending on the event types recorded by the event monitor.

For file and pipe event monitors, event records may be generated for any connection and may therefore appear in mixed order in the stream. This means that you may get a transaction event for Connection 1, immediately followed by a connection event for Connection 2. However, records belonging to a single connection or a single event will appear in their logical order. For example, a statement record (end of statement) always precedes a transaction record (end of UOW), if any. Similarly, a deadlock event record always precedes the deadlocked connection event records for each connection involved in the deadlock. The **application id** or **application handle (agent_id)** can be used to match records with a connection.

Connection header events are normally written for each connection to the database. For `deadlocks with details` event monitors, they are only written when the deadlock occurs. In this case, connection header events are only written for participants in the deadlock and not for all connections to the database.

The logical data groupings are ordered according to four different levels: Monitor, Prolog, Contents, and Epilog. Following are detailed descriptions for each level, including the corresponding event types and logical data groups.

Monitor

Information at the Monitor level is generated for all event monitors. It consists of event monitor metadata.

Table 59. Event Monitor Data Stream: Monitor Section

Event type	Logical data group	Available information
Monitor Level	event_log_stream_header	Identifies the version level and byte order of the event monitor. Applications can use this header to determine whether they can handle the evmon output stream.

Prolog

The Prolog information is generated when the event monitor is activated.

Table 60. Event Monitor Data Stream: Prolog Section

Event type	Logical data group	Available information
Log Header	event_log_header	Characteristics of the trace, for example server type and memory layout.
Database Header	event_db_header	Database name, path and activation time.
Event Monitor Start	event_start	Time when the monitor was started or restarted.
Connection Header	event_connheader	One for each current connection, includes connection time and application name. Event connection headers are only generated for connection, statement, transaction, and deadlock event monitors. Deadlocks with details event monitors produce connection headers only when a deadlock occurs.

Contents

Information specific to the event monitor's specified event types is presented in the Contents section.

Table 61. Event Monitor Data Stream: Contents Section

Event type	Logical data group	Available information
Statement Event	event_stmt	Statement level data, including text for dynamic statements. Statement event monitors do not log fetches.
Subsection Event	event_subsection	Subsection level data.
Transaction Event[1]	event_xact	Transaction level data.
Connection Event	event_conn	Connection level data.

Table 61. Event Monitor Data Stream: Contents Section (continued)

Event type	Logical data group	Available information
Deadlock Event	event_deadlock	Deadlock level data.
Deadlocked Connection Event	event_dlconn	One for each connection involved in the deadlock, includes applications involved and locks in contention.
Deadlocked Connection Event with Details	event_detailed_dlconn, lock	One for each connection involved in the deadlock, includes applications involved, locks in contention, current statement information, and other locks held by the application contention.
Overflow	event_overflow	Number of records lost - generated when writer cannot keep up with a (non-blocked) event monitor.
Deadlocks with details history[2]	event_stmt_history	List of statements executed in any unit of work that was involved in a deadlock.
Deadlocks with details history values[2]	event_data_value	Parameter markers for a statement in the event_stmt_history list.
Activities	event_activity	List of activities that completed executing on the system or were captured before completion.
	event_activitystmt	Information about the statement the activity was executing if the activity type was a statement.
	event_activityvals	The data values used as input variables for each activity that is an SQL statement. These data values do not include LOB data, long data, or structured type data.
Statistics	event_scstats	Statistics computed from the activities that executed within each service class, work class, or workload in the system, as well as statistics computed from the threshold queues.
	event_wcstats	
	event_wlstats	
	event_qstats	
	event_histogrambin	
Threshold violations	event_thresholdviolations	Information identifying the threshold violated and the time of violation.

[1] This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR UNIT OF WORK statement to monitor transaction events.

[2] This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Epilog

The Epilog information is generated during database deactivation (last application finished disconnecting):

Table 62. Event Monitor Data Stream: Epilog Section

Event type	Logical data group	Available information
Database Event	event_db	Database manager level data.
Buffer Pool Event	event_bufferpool	Buffer pool level data.
Table Space Event	event_tablespace	Table space level data.
Table Event	event_table	Table level data.

Event monitor logical data groups and monitor elements

The following table lists the logical data groupings and monitor elements that can be returned by event monitoring.

- "event_activity logical data group" on page 458
- "event_activitymetrics logical data group" on page 460 (in Version 9.7 Fix Pack 4 and later fix packs)
- "event_activitystmt logical data group" on page 462
- "event_activityvals logical data group" on page 463
- "event_bufferpool logical data group" on page 463
- "event_conn logical data group" on page 464
- "event_connheader logical data group" on page 466
- "event_connmemuse logical data group" on page 467
- "event_data_value logical data group" on page 467
- "event_db logical data group" on page 467
- "event_dbheader logical data group" on page 471
- "event_dbmemuse logical data group" on page 471
- "event_deadlock logical data group" on page 471
- "event_detailed_dlconn logical data group" on page 471
- "event_dlconn logical data group" on page 472
- "event_histogrambin logical data group" on page 473
- "event_log_header logical data group" on page 473
- "event_overflow logical data group" on page 473
- "event_qstats logical data group" on page 473
- "event_scmetrics logical data group" on page 474
- "event_scstats logical data group" on page 478
- "event_start logical data group" on page 480
- "event_stmt logical data group" on page 480
- "event_stmt_history logical data group" on page 481
- "event_subsection logical data group" on page 481
- "event_table logical data group" on page 482
- "event_tablespace logical data group" on page 482
- "event_thresholdviolations logical data group" on page 483

- "event_wlmetrics logical data group" on page 484
- "event_wlstats logical data group" on page 488
- "event_wcstats logical data group" on page 489
- "event_xact logical data group" on page 490
- "lock logical data group" on page 491
- "sqlca logical data group" on page 491

event_activity logical data group

"act_exec_time - Activity execution time monitor element" on page 497

"activate_timestamp - Activate timestamp monitor element" on page 501

"activity_id - Activity ID monitor element" on page 502

"activity_secondary_id - Activity secondary ID monitor element" on page 503

"activity_type - Activity type monitor element" on page 504

"address - IP address from which the connection was initiated" on page 506

"agent_id - Application handle (agent ID) monitor element" on page 506

"appl_id - Application ID monitor element" on page 519

"appl_name - Application name monitor element" on page 523

"arm_correlator - Application response measurement correlator monitor element" on page 530

"coord_partition_num - Coordinator partition number monitor element" on page 591

"db_work_action_set_id - Database work action set ID monitor element" on page 611

"db_work_class_id - Database work class ID monitor element" on page 611

details_xml (This XML document is a metrics document of type activity_metrics, as described in the XML schema document `sqllib/misc/DB2MonCommon.xsd`. In Version 9.7 Fix Pack 4 and later fix packs, you can also access the metrics reported in this document through the event_activitymetrics logical data group.)

"parent_activity_id - Parent activity ID monitor element" on page 797

"parent_uow_id - Parent unit of work ID monitor element" on page 797

"partial_record - Partial Record monitor element" on page 798

"pool_data_l_reads - Buffer pool data logical reads monitor element" on page 816

"pool_data_p_reads - Buffer pool data physical reads monitor element" on page 818

"num_remaps - Number of remaps monitor element" on page 780

"sc_work_action_set_id - Service class work action set ID monitor element" on page 898

"sc_work_class_id - Service class work class ID monitor element" on page 899

"section_actuals - Section actuals monitor element" on page 900

"service_subclass_name - Service subclass name monitor element" on page 908

"service_superclass_name - Service superclass name monitor element" on page 909

"session_auth_id - Session authorization ID monitor element" on page 910

"sort_overflows - Sort overflows monitor element" on page 915

"sqlca - SQL Communications Area (SQLCA)" on page 920

"time_completed - Time completed monitor element" on page 985
"time_created - Time created monitor element" on page 986
"time_started - Time started monitor element" on page 986
"total_sort_time - Total sort time monitor element" on page 1035
"total_sorts - Total sorts monitor element" on page 1036
"tpmon_acc_str - TP monitor client accounting string monitor element" on page 1039
"tpmon_client_app - TP monitor client application name monitor element" on page 1040
"tpmon_client_userid - TP monitor client user ID monitor element" on page 1040
"tpmon_client_wkstn - TP monitor client workstation name monitor element" on page 1041
"uow_id - Unit of work ID monitor element" on page 1048
"workload_id - Workload ID monitor element" on page 1063
"workload_occurrence_id - Workload occurrence identifier monitor element" on page 1065
"pool_index_l_reads - Buffer pool index logical reads monitor element" on page 825
"pool_index_p_reads - Buffer pool index physical reads monitor element" on page 827
"pool_temp_data_l_reads - Buffer pool temporary data logical reads monitor element" on page 835
"pool_temp_data_p_reads - Buffer pool temporary data physical reads monitor element" on page 837
"pool_temp_index_l_reads - Buffer pool temporary index logical reads monitor element" on page 839
"pool_temp_index_p_reads - Buffer pool temporary index physical reads monitor element" on page 841
"pool_temp_xda_l_reads - Buffer pool temporary XDA data logical reads monitor element" on page 843
"pool_temp_xda_p_reads - Buffer pool temporary XDA data physical reads monitor element" on page 845
"pool_xda_l_reads - Buffer pool XDA data logical reads monitor element" on page 849
"pool_xda_p_reads - Buffer pool XDA data physical reads monitor element" on page 851
"prep_time - Preparation time monitor element" on page 860
"query_card_estimate - Query Number of Rows Estimate" on page 868
"query_cost_estimate - Query cost estimate monitor element" on page 869
"rows_fetched - Rows fetched monitor element" on page 889
"rows_modified - Rows modified monitor element" on page 890
"rows_returned - Rows returned monitor element" on page 893
"system_cpu_time - System CPU time monitor element" on page 950
"user_cpu_time - User CPU time monitor element" on page 1053
"wl_work_action_set_id - Workload work action set identifier monitor element" on page 1058

"wl_work_class_id - Workload work class identifier monitor element" on page 1059

event_activitymetrics logical data group

This logical data group is available in Version 9.7 Fix Pack 4 and later fix packs.

"audit_events_total - Total audit events monitor element" on page 531

"audit_file_writes_total - Total audit files written monitor element" on page 533

"audit_subsystem_wait_time - Audit subsystem wait time monitor element" on page 534

"audit_subsystem_waits_total - Total audit subsystem waits monitor element" on page 536

"coord_stmt_exec_time - Execution time for statement by coordinator agent monitor element" on page 591

"deadlocks - Deadlocks detected monitor element" on page 614

"diaglog_write_wait_time - Diagnostic log file write wait time monitor element" on page 618

"diaglog_writes_total - Total diagnostic log file writes monitor element" on page 619

"direct_read_reqs - Direct read requests monitor element" on page 620

"direct_read_time - Direct read time monitor element" on page 622

"direct_reads - Direct reads from database monitor element" on page 624

"direct_write_reqs - Direct write requests monitor element" on page 626

"direct_write_time - Direct write time monitor element" on page 627

"direct_writes - Direct writes to database monitor element" on page 629

"fcm_message_recv_volume - FCM message received volume monitor element" on page 640

"fcm_message_recv_wait_time - FCM message received wait time monitor element" on page 641

"fcm_message_recvs_total - Total FCM message receives monitor element" on page 643

"fcm_message_send_volume - FCM message send volume monitor element" on page 644

"fcm_message_send_wait_time - FCM message send wait time monitor element" on page 645

"fcm_message_sends_total - Total FCM message sends monitor element" on page 646

"fcm_recv_volume - FCM received volume monitor element" on page 647

"fcm_recv_wait_time - FCM received wait time monitor element" on page 648

"fcm_recvs_total - FCM receives total monitor element" on page 650

"fcm_send_volume - FCM send volume monitor element" on page 651

"fcm_send_wait_time - FCM send wait time monitor element" on page 652

"fcm_sends_total - FCM sends total monitor element" on page 654

"fcm_tq_recv_volume - FCM table queue received volume monitor element" on page 655

"fcm_tq_recv_wait_time - FCM table queue received wait time monitor element" on page 656

"fcm_tq_recvs_total - FCM table queue receives total monitor element" on page 657

"fcm_tq_send_volume - FCM table queue send volume monitor element" on page 658
"fcm_tq_send_wait_time - FCM table queue send wait time monitor element" on page 659
"fcm_tq_sends_total - FCM table queue send total monitor element" on page 660
"lock_escals - Number of lock escalations monitor element" on page 721
"lock_timeouts - Number of lock timeouts monitor element" on page 732
"lock_wait_time - Time waited on locks monitor element" on page 734
"lock_waits - Lock waits monitor element" on page 737
"log_buffer_wait_time - Log buffer wait time monitor element" on page 740
"log_disk_wait_time - Log disk wait time monitor element" on page 741
"log_disk_waits_total - Total log disk waits monitor element" on page 743
"num_log_buffer_full - Number of times full log buffer caused agents to wait monitor element" on page 775
"num_lw_thresh_exceeded - Number of lock wait thresholds exceeded monitor element" on page 778
"pool_data_l_reads - Buffer pool data logical reads monitor element" on page 816
"pool_data_p_reads - Buffer pool data physical reads monitor element" on page 818
"pool_data_writes - Buffer pool data writes monitor element" on page 820
"pool_index_l_reads - Buffer pool index logical reads monitor element" on page 825
"pool_index_p_reads - Buffer pool index physical reads monitor element" on page 827
"pool_index_writes - Buffer pool index writes monitor element" on page 829
"pool_read_time - Total buffer pool physical read time monitor element" on page 833
"pool_temp_data_l_reads - Buffer pool temporary data logical reads monitor element" on page 835
"pool_temp_data_p_reads - Buffer pool temporary data physical reads monitor element" on page 837
"pool_temp_index_l_reads - Buffer pool temporary index logical reads monitor element" on page 839
"pool_temp_index_p_reads - Buffer pool temporary index physical reads monitor element" on page 841
"pool_temp_xda_l_reads - Buffer pool temporary XDA data logical reads monitor element" on page 843
"pool_temp_xda_p_reads - Buffer pool temporary XDA data physical reads monitor element" on page 845
"pool_write_time - Total buffer pool physical write time monitor element" on page 847
"pool_xda_l_reads - Buffer pool XDA data logical reads monitor element" on page 849
"pool_xda_p_reads - Buffer pool XDA data physical reads monitor element" on page 851
"pool_xda_writes - Buffer pool XDA data writes monitor element" on page 853

"post_shrthreshold_sorts - Post shared threshold sorts monitor element" on page 856

"post_threshold_sorts - Post threshold sorts monitor element" on page 858

"rows_modified - Rows modified monitor element" on page 890

"rows_read - Rows read monitor element" on page 892

"rows_returned - Rows returned monitor element" on page 893

"sort_overflows - Sort overflows monitor element" on page 915

"stmt_exec_time - Statement execution time monitor element" on page 931

"thresh_violations - Number of threshold violations monitor element" on page 981

"total_act_time - Total activity time monitor element" on page 988

"total_act_wait_time - Total activity wait time monitor element" on page 989

"total_app_section_executions - Total application section executions monitor element" on page 993

"total_cpu_time - Total CPU time monitor element" on page 1001

"total_routine_invocations - Total routine invocations monitor elements" on page 1016

"total_routine_non_sect_proc_time - Non-section processing time monitor element" on page 1017

"total_routine_non_sect_time - Non-section routine execution time monitor elements" on page 1017

"total_routine_time - Total routine time monitor element" on page 1018

"total_routine_user_code_proc_time - Total routine user code processing time monitor element" on page 1020

"total_routine_user_code_time - Total routine user code time monitor element" on page 1021

"total_section_proc_time - Total section processing time monitor element" on page 1027

"total_section_sort_proc_time - Total section sort processing time monitor element" on page 1028

"total_section_sort_time - Total section sort time monitor element" on page 1030

"total_section_sorts - Total section sorts monitor element" on page 1032

"total_section_time - Total section time monitor element" on page 1033

"total_sorts - Total sorts monitor element" on page 1036

"tq_tot_send_spills - Total number of table queue buffers overflowed monitor element" on page 1044

"wlm_queue_assignments_total - Workload manager total queue assignments monitor element" on page 1059

"wlm_queue_time_total - Workload manager total queue time monitor element" on page 1060

event_activitystmt logical data group

"activate_timestamp - Activate timestamp monitor element" on page 501

"activity_id - Activity ID monitor element" on page 502

"activity_secondary_id - Activity secondary ID monitor element" on page 503

"appl_id - Application ID monitor element" on page 519

"comp_env_desc - Compilation environment monitor element" on page 565

"creator - Application Creator" on page 601

"eff_stmt_text - Effective statement text monitor element" on page 633
"executable_id - Executable ID monitor element" on page 639
"package_name - Package name monitor element" on page 791
"package_version_id - Package version monitor element" on page 792
"routine_id - Routine ID monitor element" on page 888
"section_env - Section environment monitor element" on page 901
"section_number - Section number monitor element" on page 901
"stmt_first_use_time - Statement first use timestamp monitor element" on page 931
"stmt_invocation_id - Statement invocation identifier monitor element" on page 932
"stmt_isolation - Statement isolation" on page 933
"stmt_last_use_time - Statement last use timestamp monitor element" on page 934
"stmt_lock_timeout - Statement lock timeout monitor element" on page 934
"stmt_nest_level - Statement nesting level monitor element" on page 935
"stmt_pkgcache_id - Statement package cache identifier monitor element" on page 937
"stmt_query_id - Statement query identifier monitor element" on page 938
"stmt_source_id - Statement source identifier" on page 939
"stmt_text - SQL statement text monitor element" on page 940
"stmt_type - Statement type monitor element" on page 942
"uow_id - Unit of work ID monitor element" on page 1048

event_activityvals logical data group

"activate_timestamp - Activate timestamp monitor element" on page 501
"activity_id - Activity ID monitor element" on page 502
"activity_secondary_id - Activity secondary ID monitor element" on page 503
"appl_id - Application ID monitor element" on page 519
"stmt_value_data - Value data" on page 944
"stmt_value_index - Value index" on page 944
"stmt_value_isnull - Value has null value monitor element" on page 945
"stmt_value_isreopt - Variable used for statement reoptimization monitor element" on page 945
"stmt_value_type - Value type monitor element" on page 946
"uow_id - Unit of work ID monitor element" on page 1048

event_bufferpool logical data group

"bp_id - Buffer pool identifier monitor element" on page 543
"bp_name - Buffer pool name monitor element" on page 543
"db_name - Database Name" on page 608
"db_path - Database Path" on page 609
"direct_read_reqs - Direct read requests monitor element" on page 620
"direct_read_time - Direct read time monitor element" on page 622
"direct_reads - Direct reads from database monitor element" on page 624
"direct_write_reqs - Direct write requests monitor element" on page 626
"direct_write_time - Direct write time monitor element" on page 627

"direct_writes - Direct writes to database monitor element" on page 629
"event_time - Event Time" on page 637
"evmon_activates - Number of Event Monitor Activations" on page 637
"evmon_flushes - Number of Event Monitor Flushes" on page 638
"files_closed - Database files closed monitor element" on page 662
"partial_record - Partial Record monitor element" on page 798
"pool_async_data_read_reqs - Buffer pool asynchronous read requests monitor element" on page 806
"pool_async_data_reads - Buffer pool asynchronous data reads monitor element" on page 807
"pool_async_data_writes - Buffer pool asynchronous data writes monitor element" on page 808
"pool_async_index_reads - Buffer pool asynchronous index reads monitor element" on page 809
"pool_async_index_writes - Buffer pool asynchronous index writes monitor element" on page 810
"pool_async_read_time - Buffer Pool Asynchronous Read Time" on page 811
"pool_async_write_time - Buffer pool asynchronous write time monitor element" on page 812
"pool_data_l_reads - Buffer pool data logical reads monitor element" on page 816
"pool_data_p_reads - Buffer pool data physical reads monitor element" on page 818
"pool_data_writes - Buffer pool data writes monitor element" on page 820
"pool_index_l_reads - Buffer pool index logical reads monitor element" on page 825
"pool_index_p_reads - Buffer pool index physical reads monitor element" on page 827
"pool_index_writes - Buffer pool index writes monitor element" on page 829
"pool_read_time - Total buffer pool physical read time monitor element" on page 833
"pool_write_time - Total buffer pool physical write time monitor element" on page 847

event_conn logical data group

"acc_curs_blk - Accepted Block Cursor Requests" on page 494
"agent_id - Application handle (agent ID) monitor element" on page 506
"appl_id - Application ID monitor element" on page 519
"appl_priority - Application Agent Priority" on page 523
"appl_priority_type - Application Priority Type" on page 524
"appl_section_inserts - Section Inserts monitor element" on page 525
"appl_section_lookups - Section Lookups" on page 525
"authority_bitmap - User authorization level monitor element" on page 537
"authority_lvl - User authorization level monitor element" on page 538
"binds_precompiles - Binds/Precompiles Attempted" on page 540
"cat_cache_inserts - Catalog cache inserts monitor element" on page 547
"cat_cache_lookups - Catalog cache lookups monitor element" on page 548
"cat_cache_overflows - Catalog Cache Overflows" on page 550

"commit_sql_stmts - Commit Statements Attempted" on page 565
"ddl_sql_stmts - Data Definition Language (DDL) SQL Statements" on page 613
"deadlocks - Deadlocks detected monitor element" on page 614
"direct_read_reqs - Direct read requests monitor element" on page 620
"direct_read_time - Direct read time monitor element" on page 622
"direct_reads - Direct reads from database monitor element" on page 624
"direct_write_reqs - Direct write requests monitor element" on page 626
"direct_write_time - Direct write time monitor element" on page 627
"direct_writes - Direct writes to database monitor element" on page 629
"disconn_time - Database Deactivation Timestamp" on page 631
"dynamic_sql_stmts - Dynamic SQL Statements Attempted" on page 632
"failed_sql_stmts - Failed Statement Operations" on page 640
"hash_join_overflows - Hash Join Overflows" on page 678
"hash_join_small_overflows - Hash Join Small Overflows" on page 679
"int_auto_rebinds - Internal Automatic Rebinds" on page 697
"int_commits - Internal commits monitor element" on page 698
"int_deadlock_rollbacks - Internal Rollbacks Due To Deadlock" on page 699
"int_rollbacks - Internal rollbacks monitor element" on page 700
"int_rows_deleted - Internal Rows Deleted" on page 702
"int_rows_inserted - Internal Rows Inserted" on page 702
"int_rows_updated - Internal Rows Updated" on page 703
"lock_escalation - Lock escalation monitor element" on page 720
"lock_timeouts - Number of lock timeouts monitor element" on page 732
"lock_wait_time - Time waited on locks monitor element" on page 734
"lock_waits - Lock waits monitor element" on page 737
"olap_func_overflows - OLAP Function Overflows monitor element" on page 782
"partial_record - Partial Record monitor element" on page 798
"int_rows_updated - Internal Rows Updated" on page 703
"pkg_cache_inserts - Package cache inserts monitor element" on page 801
"pkg_cache_lookups - Package cache lookups monitor element" on page 802
"pool_data_l_reads - Buffer pool data logical reads monitor element" on page 816
"pool_data_p_reads - Buffer pool data physical reads monitor element" on page 818
"pool_data_writes - Buffer pool data writes monitor element" on page 820
"pool_index_l_reads - Buffer pool index logical reads monitor element" on page 825
"pool_index_p_reads - Buffer pool index physical reads monitor element" on page 827
"pool_index_writes - Buffer pool index writes monitor element" on page 829
"pool_read_time - Total buffer pool physical read time monitor element" on page 833
"pool_temp_data_l_reads - Buffer pool temporary data logical reads monitor element" on page 835

"pool_temp_data_p_reads - Buffer pool temporary data physical reads monitor element" on page 837
"pool_temp_index_l_reads - Buffer pool temporary index logical reads monitor element" on page 839
"pool_temp_index_p_reads - Buffer pool temporary index physical reads monitor element" on page 841
"pool_write_time - Total buffer pool physical write time monitor element" on page 847
"prefetch_wait_time - Time waited for prefetch monitor element" on page 860
"priv_workspace_num_overflows - Private Workspace Overflows" on page 862
"priv_workspace_section_inserts - Private Workspace Section Inserts" on page 863
"priv_workspace_section_lookups - Private Workspace Section Lookups" on page 863
"priv_workspace_size_top - Maximum Private Workspace Size" on page 864
"rej_curs_blk - Rejected Block Cursor Requests" on page 875
"rollback_sql_stmts - Rollback Statements Attempted" on page 886
"rows_read - Rows read monitor element" on page 892
"rows_selected - Rows Selected" on page 895
"rows_written - Rows Written" on page 897
"select_sql_stmts - Select SQL Statements Executed" on page 902
"sequence_no - Sequence number monitor element" on page 904
"shr_workspace_num_overflows - Shared Workspace Overflows" on page 911
"shr_workspace_section_inserts - Shared Workspace Section Inserts" on page 911
"shr_workspace_section_lookups - Shared Workspace Section Lookups" on page 912
"shr_workspace_size_top - Maximum Shared Workspace Size" on page 912
"sort_overflows - Sort overflows monitor element" on page 915
"static_sql_stmts - Static SQL Statements Attempted" on page 927
"system_cpu_time - System CPU time monitor element" on page 950
"total_hash_joins - Total Hash Joins" on page 1002
"total_hash_loops - Total Hash Loops" on page 1003
"total_olap_funcs - Total OLAP Functions monitor element" on page 1010
"total_sec_cons - Secondary Connections" on page 1027
"total_sort_time - Total sort time monitor element" on page 1035
"total_sorts - Total sorts monitor element" on page 1036
"uid_sql_stmts - Update/Insert/Delete SQL Statements Executed" on page 1046
"unread_prefetch_pages - Unread prefetch pages monitor element" on page 1047
"user_cpu_time - User CPU time monitor element" on page 1053
"x_lock_escals - Exclusive lock escalations monitor element" on page 1066
"xquery_stmts - XQuery Statements Attempted" on page 1068

event_connheader logical data group

"agent_id - Application handle (agent ID) monitor element" on page 506
"appl_id - Application ID monitor element" on page 519

"appl_name - Application name monitor element" on page 523
"auth_id - Authorization ID" on page 537
"client_db_alias - Database Alias Used by Application" on page 556
"client_pid - Client process ID monitor element" on page 558
"client_platform - Client operating platform monitor element" on page 559
"client_prdid - Client product and version ID monitor element" on page 560
"client_protocol - Client communication protocol monitor element" on page 561
"codepage_id - ID of Code Page Used by Application" on page 564
"conn_time - Time of database connection monitor element" on page 578
"corr_token - DRDA Correlation Token" on page 591
"execution_id - User Login ID" on page 639
"node_number - Node Number" on page 770
"sequence_no - Sequence number monitor element" on page 904
"territory_code - Database Territory Code" on page 981

event_connmemuse logical data group

"node_number - Node Number" on page 770
"pool_config_size - Configured Size of Memory Pool" on page 815
"pool_cur_size - Current Size of Memory Pool" on page 815
"pool_id - Memory Pool Identifier" on page 824
"pool_secondary_id - Memory Pool Secondary Identifier" on page 834
"pool_watermark - Memory Pool Watermark" on page 847

event_data_value logical data group

"deadlock_id - Deadlock Event Identifier" on page 613
"deadlock_node - Partition Number Where Deadlock Occurred" on page 614
"evmon_activates - Number of Event Monitor Activations" on page 637
"participant_no - Participant within Deadlock" on page 798
"stmt_history_id - Statement history identifier" on page 932
"stmt_value_data - Value data" on page 944
"stmt_value_index - Value index" on page 944
"stmt_value_isnull - Value has null value monitor element" on page 945
"stmt_value_isreopt - Variable used for statement reoptimization monitor element" on page 945
"stmt_value_type - Value type monitor element" on page 946

event_db logical data group

"active_hash_joins - Active hash joins" on page 501
"appl_section_inserts - Section Inserts monitor element" on page 525
"appl_section_lookups - Section Lookups" on page 525
"async_runstats – Total number of asynchronous RUNSTATS requests monitor element" on page 530
"binds_precompiles - Binds/Precompiles Attempted" on page 540
"blocks_pending_cleanup - Pending cleanup rolled-out blocks monitor element" on page 542
"cat_cache_inserts - Catalog cache inserts monitor element" on page 547
"cat_cache_lookups - Catalog cache lookups monitor element" on page 548

"cat_cache_overflows - Catalog Cache Overflows" on page 550
"cat_cache_size_top - Catalog cache high watermark monitor element" on page 551
"catalog_node - Catalog Node Number" on page 551
"catalog_node_name - Catalog Node Network Name" on page 552
"commit_sql_stmts - Commit Statements Attempted" on page 565
"connections_top - Maximum Number of Concurrent Connections" on page 580
"db_heap_top - Maximum Database Heap Allocated" on page 607
"ddl_sql_stmts - Data Definition Language (DDL) SQL Statements" on page 613
"deadlocks - Deadlocks detected monitor element" on page 614
"direct_read_reqs - Direct read requests monitor element" on page 620
"direct_read_time - Direct read time monitor element" on page 622
"direct_reads - Direct reads from database monitor element" on page 624
"direct_write_reqs - Direct write requests monitor element" on page 626
"direct_write_time - Direct write time monitor element" on page 627
"direct_writes - Direct writes to database monitor element" on page 629
"disconn_time - Database Deactivation Timestamp" on page 631
"dynamic_sql_stmts - Dynamic SQL Statements Attempted" on page 632
"evmon_activates - Number of Event Monitor Activations" on page 637
"evmon_flushes - Number of Event Monitor Flushes" on page 638
"failed_sql_stmts - Failed Statement Operations" on page 640
"files_closed - Database files closed monitor element" on page 662
"hash_join_overflows - Hash Join Overflows" on page 678
"hash_join_small_overflows - Hash Join Small Overflows" on page 679
"int_auto_rebinds - Internal Automatic Rebinds" on page 697
"int_commits - Internal commits monitor element" on page 698
"int_rollbacks - Internal rollbacks monitor element" on page 700
"int_rows_deleted - Internal Rows Deleted" on page 702
"int_rows_inserted - Internal Rows Inserted" on page 702
"int_rows_updated - Internal Rows Updated" on page 703
"lock_escals - Number of lock escalations monitor element" on page 721
"lock_timeouts - Number of lock timeouts monitor element" on page 732
"lock_wait_time - Time waited on locks monitor element" on page 734
"lock_waits - Lock waits monitor element" on page 737
"log_held_by_dirty_pages - Amount of Log Space Accounted for by Dirty Pages" on page 744
"log_read_time - Log Read Time" on page 744
"log_reads - Number of Log Pages Read" on page 745
"log_to_redo_for_recovery - Amount of Log to be Redone for Recovery" on page 745
"log_write_time - Log Write Time" on page 746
"log_writes - Number of Log Pages Written" on page 746
"num_log_read_io - Number of Log Reads" on page 778
"num_log_write_io - Number of Log Writes" on page 778
"num_threshold_violations - Number of threshold violations monitor element" on page 780

"olap_func_overflows - OLAP Function Overflows monitor element" on page 782
"partial_record - Partial Record monitor element" on page 798
"pkg_cache_inserts - Package cache inserts monitor element" on page 801
"pkg_cache_lookups - Package cache lookups monitor element" on page 802
"pkg_cache_num_overflows - Package Cache Overflows" on page 804
"pkg_cache_size_top - Package cache high watermark" on page 805
"pool_async_data_read_reqs - Buffer pool asynchronous read requests monitor element" on page 806
"pool_async_data_reads - Buffer pool asynchronous data reads monitor element" on page 807
"pool_async_data_writes - Buffer pool asynchronous data writes monitor element" on page 808
"pool_async_index_read_reqs - Buffer pool asynchronous index read requests monitor element" on page 808
"pool_async_index_reads - Buffer pool asynchronous index reads monitor element" on page 809
"pool_async_index_writes - Buffer pool asynchronous index writes monitor element" on page 810
"pool_async_read_time - Buffer Pool Asynchronous Read Time" on page 811
"pool_async_write_time - Buffer pool asynchronous write time monitor element" on page 812
"pool_data_l_reads - Buffer pool data logical reads monitor element" on page 816
"pool_data_p_reads - Buffer pool data physical reads monitor element" on page 818
"pool_data_writes - Buffer pool data writes monitor element" on page 820
"pool_drty_pg_steal_clns - Buffer pool victim page cleaners triggered monitor element" on page 822
"pool_drty_pg_thrsh_clns - Buffer pool threshold cleaners triggered monitor element" on page 823
"pool_index_l_reads - Buffer pool index logical reads monitor element" on page 825
"pool_index_p_reads - Buffer pool index physical reads monitor element" on page 827
"pool_index_writes - Buffer pool index writes monitor element" on page 829
"pool_lsn_gap_clns - Buffer pool log space cleaners triggered monitor element" on page 831
"pool_no_victim_buffer - Buffer pool no victim buffers monitor element" on page 832
"pool_read_time - Total buffer pool physical read time monitor element" on page 833
"pool_temp_data_l_reads - Buffer pool temporary data logical reads monitor element" on page 835
"pool_temp_data_p_reads - Buffer pool temporary data physical reads monitor element" on page 837
"pool_temp_index_l_reads - Buffer pool temporary index logical reads monitor element" on page 839

"pool_temp_index_p_reads - Buffer pool temporary index physical reads monitor element" on page 841
"pool_write_time - Total buffer pool physical write time monitor element" on page 847
"post_shrthreshold_hash_joins - Post threshold hash joins" on page 855
"post_shrthreshold_sorts - Post shared threshold sorts monitor element" on page 856
"prefetch_wait_time - Time waited for prefetch monitor element" on page 860
"priv_workspace_num_overflows - Private Workspace Overflows" on page 862
"priv_workspace_section_inserts - Private Workspace Section Inserts" on page 863
"priv_workspace_section_lookups - Private Workspace Section Lookups" on page 863
"priv_workspace_size_top - Maximum Private Workspace Size" on page 864
"rollback_sql_stmts - Rollback Statements Attempted" on page 886
"rows_deleted - Rows deleted monitor element" on page 889
"rows_inserted - Rows inserted monitor element" on page 890
"rows_read - Rows read monitor element" on page 892
"rows_selected - Rows Selected" on page 895
"rows_updated - Rows updated monitor element" on page 896
"sec_log_used_top - Maximum Secondary Log Space Used" on page 899
"select_sql_stmts - Select SQL Statements Executed" on page 902
"server_platform - Server Operating System" on page 906
"shr_workspace_num_overflows - Shared Workspace Overflows" on page 911
"shr_workspace_section_inserts - Shared Workspace Section Inserts" on page 911
"shr_workspace_section_lookups - Shared Workspace Section Lookups" on page 912
"shr_workspace_size_top - Maximum Shared Workspace Size" on page 912
"sort_overflows - Sort overflows monitor element" on page 915
"static_sql_stmts - Static SQL Statements Attempted" on page 927
"stats_cache_size – Size of statistics cache monitor element" on page 928
"stats_fabricate_time – Total time spent on statistics fabrication activities monitor element" on page 929
"stats_fabrications – Total number of statistics fabrications monitor elements" on page 929
"sync_runstats – Total number of synchronous RUNSTATS activities monitor element" on page 948
"sync_runstats_time – Total time spent on synchronous RUNSTATS activities monitor element" on page 949
"tot_log_used_top - Maximum Total Log Space Used" on page 987
"total_cons - Connects Since Database Activation" on page 1000
"total_hash_joins - Total Hash Joins" on page 1002
"total_hash_loops - Total Hash Loops" on page 1003
"total_olap_funcs - Total OLAP Functions monitor element" on page 1010
"total_sort_time - Total sort time monitor element" on page 1035
"total_sorts - Total sorts monitor element" on page 1036

"uid_sql_stmts - Update/Insert/Delete SQL Statements Executed" on page 1046
"unread_prefetch_pages - Unread prefetch pages monitor element" on page 1047
"x_lock_escals - Exclusive lock escalations monitor element" on page 1066
"xquery_stmts - XQuery Statements Attempted" on page 1068

event_dbheader logical data group

"conn_time - Time of database connection monitor element" on page 578
"db_name - Database Name" on page 608
"db_path - Database Path" on page 609

event_dbmemuse logical data group

"node_number - Node Number" on page 770
"pool_config_size - Configured Size of Memory Pool" on page 815
"pool_cur_size - Current Size of Memory Pool" on page 815
"pool_id - Memory Pool Identifier" on page 824
"pool_watermark - Memory Pool Watermark" on page 847

event_deadlock logical data group

"deadlock_id - Deadlock Event Identifier" on page 613
"deadlock_node - Partition Number Where Deadlock Occurred" on page 614
"dl_conns - Connections involved in deadlock monitor element" on page 632
"evmon_activates - Number of Event Monitor Activations" on page 637
"rolled_back_agent_id - Rolled Back Agent" on page 886
"rolled_back_appl_id - Rolled Back Application" on page 887
"rolled_back_participant_no - Rolled back application participant monitor element" on page 887
"rolled_back_sequence_no - Rolled Back Sequence Number" on page 888
"start_time - Event Start Time" on page 927

event_detailed_dlconn logical data group

"agent_id - Application handle (agent ID) monitor element" on page 506
"appl_id - Application ID monitor element" on page 519
"appl_id_holding_lk - Application ID Holding Lock" on page 521
"blocking_cursor - Blocking Cursor" on page 542
"consistency_token - Package consistency token monitor element" on page 580
"creator - Application Creator" on page 601
"cursor_name - Cursor Name" on page 603
"data_partition_id - Data partition identifier monitor element" on page 604
"deadlock_id - Deadlock Event Identifier" on page 613
"deadlock_node - Partition Number Where Deadlock Occurred" on page 614
"evmon_activates - Number of Event Monitor Activations" on page 637
"lock_escalation - Lock escalation monitor element" on page 720
"lock_mode - Lock mode monitor element" on page 724
"lock_mode_requested - Lock mode requested monitor element" on page 725
"lock_node - Lock Node" on page 727
"lock_object_name - Lock Object Name" on page 727

"lock_object_type - Lock object type waited on monitor element" on page 728
"lock_wait_start_time - Lock wait start timestamp monitor element" on page 734
"locks_held - Locks held monitor element" on page 738
"locks_in_list - Number of Locks Reported" on page 740
"package_name - Package name monitor element" on page 791
"package_version_id - Package version monitor element" on page 792
"participant_no - Participant within Deadlock" on page 798
"participant_no_holding_lk - Participant Holding a Lock on the Object Required by Application" on page 799
"section_number - Section number monitor element" on page 901
"sequence_no - Sequence number monitor element" on page 904
"sequence_no_holding_lk - Sequence Number Holding Lock" on page 904
"start_time - Event Start Time" on page 927
"stmt_operation/operation - Statement operation monitor element" on page 935
"stmt_text - SQL statement text monitor element" on page 940
"stmt_type - Statement type monitor element" on page 942
"table_name - Table name monitor element" on page 952
"table_schema - Table schema name monitor element" on page 953
"tablespace_name - Table space name monitor element" on page 961

event_dlconn logical data group

"agent_id - Application handle (agent ID) monitor element" on page 506
"appl_id - Application ID monitor element" on page 519
"appl_id_holding_lk - Application ID Holding Lock" on page 521
"data_partition_id - Data partition identifier monitor element" on page 604
"deadlock_id - Deadlock Event Identifier" on page 613
"deadlock_node - Partition Number Where Deadlock Occurred" on page 614
"evmon_activates - Number of Event Monitor Activations" on page 637
"lock_attributes - Lock attributes monitor element" on page 717
"lock_count - Lock count monitor element" on page 718
"lock_current_mode - Original lock mode before conversion monitor element" on page 719
"lock_escalation - Lock escalation monitor element" on page 720
"lock_hold_count - Lock hold count monitor element" on page 723
"lock_mode - Lock mode monitor element" on page 724
"lock_mode_requested - Lock mode requested monitor element" on page 725
"lock_name - Lock name monitor element" on page 726
"lock_node - Lock Node" on page 727
"lock_object_name - Lock Object Name" on page 727
"lock_object_type - Lock object type waited on monitor element" on page 728
"lock_release_flags - Lock release flags monitor element" on page 730
"lock_wait_start_time - Lock wait start timestamp monitor element" on page 734
"participant_no - Participant within Deadlock" on page 798

"participant_no_holding_lk - Participant Holding a Lock on the Object Required by Application" on page 799

"sequence_no - Sequence number monitor element" on page 904

"sequence_no_holding_lk - Sequence Number Holding Lock" on page 904

"start_time - Event Start Time" on page 927

"table_name - Table name monitor element" on page 952

"table_schema - Table schema name monitor element" on page 953

"tablespace_name - Table space name monitor element" on page 961

"tpmon_acc_str - TP monitor client accounting string monitor element" on page 1039

"tpmon_client_app - TP monitor client application name monitor element" on page 1040

"tpmon_client_userid - TP monitor client user ID monitor element" on page 1040

"tpmon_client_wkstn - TP monitor client workstation name monitor element" on page 1041

event_histogrambin logical data group

"bin_id - Histogram bin identifier monitor element" on page 540

"bottom - Histogram bin bottom monitor element" on page 542

"histogram_type - Histogram type monitor element" on page 679

"number_in_bin - Number in bin monitor element" on page 782

"service_class_id - Service class ID monitor element" on page 907

"statistics_timestamp - Statistics timestamp monitor element" on page 928

"top - Histogram bin top monitor element" on page 987

"work_action_set_id - Work action set ID monitor element" on page 1062

"work_class_id - Work class ID monitor element" on page 1063

"workload_id - Workload ID monitor element" on page 1063

event_log_header logical data group

"byte_order - Byte Order of Event Data" on page 547

"codepage_id - ID of Code Page Used by Application" on page 564

"event_monitor_name - Event Monitor Name" on page 637

"num_nodes_in_db2_instance - Number of Nodes in Partition" on page 780

"server_instance_name - Server Instance Name" on page 905

"server_prdid - Server Product/Version ID" on page 906

"territory_code - Database Territory Code" on page 981

"version - Version of Monitor Data" on page 1057

event_overflow logical data group

"count - Number of Event Monitor Overflows" on page 592

"first_overflow_time - Time of First Event Overflow" on page 663

"last_overflow_time - Time of Last Event Overflow" on page 712

"node_number - Node Number" on page 770

event_qstats logical data group

"last_wlm_reset - Time of last reset monitor element" on page 714

"queue_assignments_total - Queue assignments total monitor element" on page 870

"queue_size_top - Queue size top monitor element" on page 870

"queue_time_total - Queue time total monitor element" on page 870

"service_subclass_name - Service subclass name monitor element" on page 908

"service_superclass_name - Service superclass name monitor element" on page 909

"statistics_timestamp - Statistics timestamp monitor element" on page 928

"threshold_domain - Threshold domain monitor element" on page 983

"threshold_name - Threshold name monitor element" on page 984

"threshold_predicate - Threshold predicate monitor element" on page 984

"thresholdid - Threshold ID monitor element" on page 985

"work_action_set_name - Work action set name monitor element" on page 1062

"work_class_name - Work class name monitor element" on page 1063

event_scmetrics logical data group

Note: Monitor elements in this logical data group are calculated to show the change in value for each monitor element since the last time statistics were collected, or since database activation, which ever happed most recently.

"wlm_queue_time_total - Workload manager total queue time monitor element" on page 1060

"wlm_queue_assignments_total - Workload manager total queue assignments monitor element" on page 1059

"fcm_tq_recv_wait_time - FCM table queue received wait time monitor element" on page 656

"fcm_message_recv_wait_time - FCM message received wait time monitor element" on page 641

"fcm_tq_send_wait_time - FCM table queue send wait time monitor element" on page 659

"fcm_message_send_wait_time - FCM message send wait time monitor element" on page 645

"agent_wait_time - Agent wait time monitor element" on page 511

"agent_waits_total - Total agent waits monitor element" on page 512

"lock_wait_time - Time waited on locks monitor element" on page 734

"lock_waits - Lock waits monitor element" on page 737

"direct_read_time - Direct read time monitor element" on page 622

"direct_read_reqs - Direct read requests monitor element" on page 620

"direct_write_time - Direct write time monitor element" on page 627

"direct_write_reqs - Direct write requests monitor element" on page 626

"log_buffer_wait_time - Log buffer wait time monitor element" on page 740

"num_log_buffer_full - Number of times full log buffer caused agents to wait monitor element" on page 775

"log_disk_wait_time - Log disk wait time monitor element" on page 741

"log_disk_waits_total - Total log disk waits monitor element" on page 743

"tcpip_recv_wait_time - TCP/IP received wait time monitor element" on page 976

"tcpip_recvs_total - TCP/IP receives total monitor element" on page 977

"client_idle_wait_time - Client idle wait time monitor element" on page 557
"ipc_recv_wait_time - Interprocess communication received wait time monitor element" on page 705
"ipc_recvs_total - Interprocess communication receives total monitor element" on page 706
"ipc_send_wait_time - Interprocess communication send wait time monitor element" on page 708
"ipc_sends_total - Interprocess communication send total monitor element" on page 709
"tcpip_send_wait_time - TCP/IP send wait time monitor element" on page 979
"tcpip_sends_total - TCP/IP sends total monitor element" on page 980
"pool_write_time - Total buffer pool physical write time monitor element" on page 847
"pool_read_time - Total buffer pool physical read time monitor element" on page 833
"audit_file_write_wait_time - Audit file write wait time monitor element" on page 532
"audit_file_writes_total - Total audit files written monitor element" on page 533
"audit_subsystem_wait_time - Audit subsystem wait time monitor element" on page 534
"audit_subsystem_waits_total - Total audit subsystem waits monitor element" on page 536
"diaglog_write_wait_time - Diagnostic log file write wait time monitor element" on page 618
"diaglog_writes_total - Total diagnostic log file writes monitor element" on page 619
"fcm_send_wait_time - FCM send wait time monitor element" on page 652
"fcm_recv_wait_time - FCM received wait time monitor element" on page 648
"total_wait_time - Total wait time monitor element" on page 1038
"rqsts_completed_total - Total requests completed monitor element" on page 897
"total_rqst_time - Total request time monitor element" on page 1023
"app_rqsts_completed_total - Total application requests completed monitor element" on page 518
"total_app_rqst_time - Total application request time monitor element" on page 992
"total_section_sort_proc_time - Total section sort processing time monitor element" on page 1028
"total_section_sorts - Total section sorts monitor element" on page 1032
"total_section_sort_time - Total section sort time monitor element" on page 1030
"rows_read - Rows read monitor element" on page 892
"rows_modified - Rows modified monitor element" on page 890
"pool_data_l_reads - Buffer pool data logical reads monitor element" on page 816
"pool_index_l_reads - Buffer pool index logical reads monitor element" on page 825
"pool_temp_data_l_reads - Buffer pool temporary data logical reads monitor element" on page 835

"pool_temp_index_l_reads - Buffer pool temporary index logical reads monitor element" on page 839

"pool_xda_l_reads - Buffer pool XDA data logical reads monitor element" on page 849

"pool_temp_xda_l_reads - Buffer pool temporary XDA data logical reads monitor element" on page 843

"total_cpu_time - Total CPU time monitor element" on page 1001

"act_completed_total - Total completed activities monitor element" on page 495

"pool_data_p_reads - Buffer pool data physical reads monitor element" on page 818

"pool_temp_data_p_reads - Buffer pool temporary data physical reads monitor element" on page 837

"pool_xda_p_reads - Buffer pool XDA data physical reads monitor element" on page 851

"pool_temp_xda_p_reads - Buffer pool temporary XDA data physical reads monitor element" on page 845

"pool_index_p_reads - Buffer pool index physical reads monitor element" on page 827

"pool_temp_index_p_reads - Buffer pool temporary index physical reads monitor element" on page 841

"pool_data_writes - Buffer pool data writes monitor element" on page 820

"pool_xda_writes - Buffer pool XDA data writes monitor element" on page 853

"pool_index_writes - Buffer pool index writes monitor element" on page 829

"direct_reads - Direct reads from database monitor element" on page 624

"direct_writes - Direct writes to database monitor element" on page 629

"rows_returned - Rows returned monitor element" on page 893

"deadlocks - Deadlocks detected monitor element" on page 614

"lock_timeouts - Number of lock timeouts monitor element" on page 732

"lock_escals - Number of lock escalations monitor element" on page 721

"fcm_sends_total - FCM sends total monitor element" on page 654

"fcm_recvs_total - FCM receives total monitor element" on page 650

"fcm_send_volume - FCM send volume monitor element" on page 651

"fcm_recv_volume - FCM received volume monitor element" on page 647

"fcm_message_sends_total - Total FCM message sends monitor element" on page 646

"fcm_message_recvs_total - Total FCM message receives monitor element" on page 643

"fcm_message_send_volume - FCM message send volume monitor element" on page 644

"fcm_message_recv_volume - FCM message received volume monitor element" on page 640

"fcm_tq_sends_total - FCM table queue send total monitor element" on page 660

"fcm_tq_recvs_total - FCM table queue receives total monitor element" on page 657

"fcm_tq_send_volume - FCM table queue send volume monitor element" on page 658

"fcm_tq_recv_volume - FCM table queue received volume monitor element" on page 655

"tq_tot_send_spills - Total number of table queue buffers overflowed monitor element" on page 1044

"tcpip_send_volume - TCP/IP send volume monitor element" on page 978

"tcpip_recv_volume - TCP/IP received volume monitor element" on page 975

"ipc_send_volume - Interprocess communication send volume monitor element" on page 707

"ipc_recv_volume - Interprocess communication received volume monitor element" on page 704

"post_threshold_sorts - Post threshold sorts monitor element" on page 858

"post_shrthreshold_sorts - Post shared threshold sorts monitor element" on page 856

"sort_overflows - Sort overflows monitor element" on page 915

"audit_events_total - Total audit events monitor element" on page 531

"total_rqst_mapped_in - Total request mapped-in monitor element" on page 1022

"total_rqst_mapped_out - Total request mapped-out monitor element" on page 1022

"act_rejected_total - Total rejected activities monitor element" on page 497

"act_aborted_total - Total aborted activities monitor element" on page 494

"total_sorts - Total sorts monitor element" on page 1036

"total_routine_time - Total routine time monitor element" on page 1018

"total_compile_proc_time - Total compile processing time monitor element" on page 998

"total_compilations - Total compilations monitor element" on page 997

"total_compile_time - Total compile time monitor element" on page 999

"total_implicit_compile_proc_time - Total implicit compile processing time monitor element" on page 1004

"total_implicit_compilations - Total implicit complications monitor element" on page 1003

"total_implicit_compile_time - Total implicit compile time monitor element" on page 1005

"total_runstats_proc_time - Total runtime statistics processing time monitor element" on page 1025

"total_runstats - Total runtime statistics monitor element" on page 1024

"total_runstats_time - Total runtime statistics time monitor element" on page 1026

"total_reorg_proc_time - Total reorganization processing time monitor element" on page 1011

"total_reorgs - Total reorganizations monitor element" on page 1013

"total_reorg_time - Total reorganization time monitor element" on page 1012

"total_load_proc_time - Total load processing time monitor element" on page 1006

"total_loads - Total loads monitor element" on page 1008

"total_load_time - Total load time monitor element" on page 1007

"total_section_proc_time - Total section processing time monitor element" on page 1027

"total_app_section_executions - Total application section executions monitor element" on page 993

"total_section_time - Total section time monitor element" on page 1033

"total_commit_proc_time - Total commits processing time monitor element" on page 995

"total_app_commits - Total application commits monitor elements" on page 990

"total_commit_time - Total commit time monitor element" on page 996

"total_rollback_proc_time - Total rollback processing time monitor element" on page 1014

"total_app_rollbacks - Total application rollbacks monitor element" on page 991

"total_rollback_time - Total rollback time monitor element" on page 1015

"total_routine_user_code_proc_time - Total routine user code processing time monitor element" on page 1020

"total_routine_user_code_time - Total routine user code time monitor element" on page 1021

"thresh_violations - Number of threshold violations monitor element" on page 981

"num_lw_thresh_exceeded - Number of lock wait thresholds exceeded monitor element" on page 778

"total_routine_invocations - Total routine invocations monitor elements" on page 1016

"int_commits - Internal commits monitor element" on page 698

"int_rollbacks - Internal rollbacks monitor element" on page 700

"cat_cache_inserts - Catalog cache inserts monitor element" on page 547

"cat_cache_lookups - Catalog cache lookups monitor element" on page 548

"pkg_cache_inserts - Package cache inserts monitor element" on page 801

"pkg_cache_lookups - Package cache lookups monitor element" on page 802

"act_rqsts_total - Total activity requests monitor elements" on page 500

"total_act_wait_time - Total activity wait time monitor element" on page 989

"total_act_time - Total activity time monitor element" on page 988

FCM_TQ_RECV_WAITS_TOTAL

FCM_MESSAGE_RECV_WAITS_TOTAL

FCM_TQ_SEND_WAITS_TOTAL

FCM_MESSAGE_SEND_WAITS_TOTAL

FCM_SEND_WAITS_TOTAL

FCM_RECV_WAITS_TOTAL

event_scstats logical data group

"act_cpu_time_top – Activity CPU time top monitor element" on page 496

"act_remapped_in – Activities remapped in monitor element" on page 498

"act_remapped_out – Activities remapped out monitor element" on page 499

"act_rows_read_top – Activity rows read top monitor element" on page 499

"agg_temp_tablespace_top - Aggregate temporary table space top monitor element" on page 516

"concurrent_act_top - Concurrent activity top monitor element" on page 568

"concurrent_wlo_top - Concurrent workload occurrences top monitor element" on page 569

"concurrent_connection_top - Concurrent connection top monitor element" on page 568

"coord_act_aborted_total - Coordinator activities aborted total monitor element" on page 583

"coord_act_completed_total - Coordinator activities completed total monitor element" on page 584

"coord_act_est_cost_avg - Coordinator activity estimated cost average monitor element" on page 584

"coord_act_exec_time_avg - Coordinator activities execution time average monitor element" on page 585

"coord_act_interarrival_time_avg - Coordinator activity arrival time average monitor element" on page 586

"coord_act_lifetime_avg - Coordinator activity lifetime average monitor element" on page 587

"coord_act_lifetime_top - Coordinator activity lifetime top monitor element" on page 587

"coord_act_queue_time_avg - Coordinator activity queue time average monitor element" on page 588

"coord_act_rejected_total - Coordinator activities rejected total monitor element" on page 589

"cost_estimate_top - Cost estimate top monitor element" on page 592

details_xml*

"last_wlm_reset - Time of last reset monitor element" on page 714

metrics* (In Version 9.7 Fix Pack 6 and later fix packs, you can also access metrics reported in this document through the "event_scmetrics logical data group" on page 474.)

"request_exec_time_avg - Request execution time average monitor element" on page 884

"rows_returned_top - Actual rows returned top monitor element" on page 895

"service_class_id - Service class ID monitor element" on page 907

"service_subclass_name - Service subclass name monitor element" on page 908

"service_superclass_name - Service superclass name monitor element" on page 909

"statistics_timestamp - Statistics timestamp monitor element" on page 928

"temp_tablespace_top - Temporary table space top monitor element" on page 980

"uow_total_time_top - UOW total time top monitor element" on page 1052

*Two XML documents are included in the output of this logical data group. In the `metrics` document, the values of the elements show the change in value for the monitor element since the last time statistics were collected. The values of the elements contained in `details_xml` are not reset at each interval; they are reset only when the database is reactivated. These XML document are metrics documents of type system_metrics, as described in the XML schema document `sqllib/misc/DB2MonCommon.xsd`.

Important: Starting with Version 9.7 Fix Pack 6, the XML document details_xml is deprecated in the statistics event monitor, and might be removed in a future release. For more information, see Reporting of metrics in details_xml by the statistics event monitor has been deprecated"Reporting of metrics in details_xml by the statistics event monitor has been deprecated" in *What's New for DB2 Version 9.7*.

event_start logical data group

"start_time - Event Start Time" on page 927

event_stmt logical data group

"agent_id - Application handle (agent ID) monitor element" on page 506
"agents_top - Number of Agents Created" on page 515
"appl_id - Application ID monitor element" on page 519
"blocking_cursor - Blocking Cursor" on page 542
"consistency_token - Package consistency token monitor element" on page 580
"creator - Application Creator" on page 601
"cursor_name - Cursor Name" on page 603
"fetch_count - Number of Successful Fetches" on page 661
"int_rows_deleted - Internal Rows Deleted" on page 702
"int_rows_inserted - Internal Rows Inserted" on page 702
"int_rows_updated - Internal Rows Updated" on page 703
"package_name - Package name monitor element" on page 791
"package_version_id - Package version monitor element" on page 792
"partial_record - Partial Record monitor element" on page 798
"pool_data_l_reads - Buffer pool data logical reads monitor element" on page 816
"pool_data_p_reads - Buffer pool data physical reads monitor element" on page 818
"pool_index_l_reads - Buffer pool index logical reads monitor element" on page 825
"pool_index_p_reads - Buffer pool index physical reads monitor element" on page 827
"pool_temp_data_l_reads - Buffer pool temporary data logical reads monitor element" on page 835
"pool_temp_data_p_reads - Buffer pool temporary data physical reads monitor element" on page 837
"pool_temp_index_l_reads - Buffer pool temporary index logical reads monitor element" on page 839
"pool_temp_index_p_reads - Buffer pool temporary index physical reads monitor element" on page 841
"rows_read - Rows read monitor element" on page 892
"rows_written - Rows Written" on page 897
"section_number - Section number monitor element" on page 901
"sequence_no - Sequence number monitor element" on page 904
"sort_overflows - Sort overflows monitor element" on page 915
"sql_req_id - Request Identifier for SQL Statement" on page 919
"sqlca - SQL Communications Area (SQLCA)" on page 920
"start_time - Event Start Time" on page 927
"stats_fabricate_time – Total time spent on statistics fabrication activities monitor element" on page 929
"stmt_operation/operation - Statement operation monitor element" on page 935
"stmt_text - SQL statement text monitor element" on page 940
"stmt_type - Statement type monitor element" on page 942

"stop_time - Event Stop Time" on page 947
"sync_runstats_time – Total time spent on synchronous RUNSTATS activities monitor element" on page 949
"system_cpu_time - System CPU time monitor element" on page 950
"total_sort_time - Total sort time monitor element" on page 1035
"total_sorts - Total sorts monitor element" on page 1036
"user_cpu_time - User CPU time monitor element" on page 1053

event_stmt_history logical data group

"comp_env_desc - Compilation environment monitor element" on page 565
"creator - Application Creator" on page 601
"deadlock_id - Deadlock Event Identifier" on page 613
"deadlock_node - Partition Number Where Deadlock Occurred" on page 614
"evmon_activates - Number of Event Monitor Activations" on page 637
"package_name - Package name monitor element" on page 791
"package_version_id - Package version monitor element" on page 792
"participant_no - Participant within Deadlock" on page 798
"section_number - Section number monitor element" on page 901
"sequence_no - Sequence number monitor element" on page 904
"stmt_first_use_time - Statement first use timestamp monitor element" on page 931
"stmt_history_id - Statement history identifier" on page 932
"stmt_invocation_id - Statement invocation identifier monitor element" on page 932
"stmt_isolation - Statement isolation" on page 933
"stmt_last_use_time - Statement last use timestamp monitor element" on page 934
"stmt_lock_timeout - Statement lock timeout monitor element" on page 934
"stmt_nest_level - Statement nesting level monitor element" on page 935
"stmt_pkgcache_id - Statement package cache identifier monitor element" on page 937
"stmt_query_id - Statement query identifier monitor element" on page 938
"stmt_source_id - Statement source identifier" on page 939
"stmt_text - SQL statement text monitor element" on page 940
"stmt_type - Statement type monitor element" on page 942

event_subsection logical data group

"agent_id - Application handle (agent ID) monitor element" on page 506
"num_agents - Number of Agents Working on a Statement" on page 771
"partial_record - Partial Record monitor element" on page 798
"ss_exec_time - Subsection Execution Elapsed Time" on page 924
"ss_node_number - Subsection Node Number" on page 925
"ss_number - Subsection number monitor element" on page 925
"ss_sys_cpu_time - System CPU Time used by Subsection" on page 926
"ss_usr_cpu_time - User CPU Time used by Subsection" on page 926
"tq_max_send_spills - Maximum number of table queue buffers overflows" on page 1042

"tq_rows_read - Number of Rows Read from table queues" on page 1043
"tq_rows_written - Number of rows written to table queues" on page 1044
"tq_tot_send_spills - Total number of table queue buffers overflowed monitor element" on page 1044

event_table logical data group

"data_object_pages - Data Object Pages" on page 603
"data_partition_id - Data partition identifier monitor element" on page 604
"event_time - Event Time" on page 637
"evmon_activates - Number of Event Monitor Activations" on page 637
"evmon_flushes - Number of Event Monitor Flushes" on page 638
"index_object_pages - Index Object Pages" on page 693
"lob_object_pages - LOB Object Pages" on page 715
"long_object_pages - Long Object Pages" on page 747
"overflow_accesses - Accesses to overflowed records monitor element" on page 789
"page_reorgs - Page reorganizations monitor element" on page 794
"partial_record - Partial Record monitor element" on page 798
"rows_read - Rows read monitor element" on page 892
"rows_written - Rows Written" on page 897
"table_name - Table name monitor element" on page 952
"table_schema - Table schema name monitor element" on page 953
"table_type - Table type monitor element" on page 954

event_tablespace logical data group

"direct_read_reqs - Direct read requests monitor element" on page 620
"direct_read_time - Direct read time monitor element" on page 622
"direct_reads - Direct reads from database monitor element" on page 624
"direct_write_reqs - Direct write requests monitor element" on page 626
"direct_write_time - Direct write time monitor element" on page 627
"direct_writes - Direct writes to database monitor element" on page 629
"event_time - Event Time" on page 637
"evmon_activates - Number of Event Monitor Activations" on page 637
"evmon_flushes - Number of Event Monitor Flushes" on page 638
"files_closed - Database files closed monitor element" on page 662
"partial_record - Partial Record monitor element" on page 798
"pool_async_data_read_reqs - Buffer pool asynchronous read requests monitor element" on page 806
"pool_async_data_reads - Buffer pool asynchronous data reads monitor element" on page 807
"pool_async_data_writes - Buffer pool asynchronous data writes monitor element" on page 808
"pool_async_index_read_reqs - Buffer pool asynchronous index read requests monitor element" on page 808
"pool_async_index_reads - Buffer pool asynchronous index reads monitor element" on page 809

"pool_async_index_writes - Buffer pool asynchronous index writes monitor element" on page 810

"pool_async_read_time - Buffer Pool Asynchronous Read Time" on page 811

"pool_async_write_time - Buffer pool asynchronous write time monitor element" on page 812

"pool_data_l_reads - Buffer pool data logical reads monitor element" on page 816

"pool_data_p_reads - Buffer pool data physical reads monitor element" on page 818

"pool_data_writes - Buffer pool data writes monitor element" on page 820

"pool_index_l_reads - Buffer pool index logical reads monitor element" on page 825

"pool_index_p_reads - Buffer pool index physical reads monitor element" on page 827

"pool_index_writes - Buffer pool index writes monitor element" on page 829

"pool_no_victim_buffer - Buffer pool no victim buffers monitor element" on page 832

"pool_read_time - Total buffer pool physical read time monitor element" on page 833

"pool_temp_data_l_reads - Buffer pool temporary data logical reads monitor element" on page 835

"pool_temp_data_p_reads - Buffer pool temporary data physical reads monitor element" on page 837

"pool_temp_index_l_reads - Buffer pool temporary index logical reads monitor element" on page 839

"pool_temp_index_p_reads - Buffer pool temporary index physical reads monitor element" on page 841

"pool_write_time - Total buffer pool physical write time monitor element" on page 847

"tablespace_name - Table space name monitor element" on page 961

event_thresholdviolations logical data group

"activate_timestamp - Activate timestamp monitor element" on page 501

"activity_collected - Activity collected monitor element" on page 502

"activity_id - Activity ID monitor element" on page 502

"agent_id - Application handle (agent ID) monitor element" on page 506

"appl_id - Application ID monitor element" on page 519

"appl_name - Application name monitor element" on page 523

"client_acctng - Client accounting string monitor element" on page 554

"client_applname - Client application name monitor element" on page 555

"client_hostname - Client hostname monitor element" on page 557

"appl_name - Application name monitor element" on page 523

"client_pid - Client process ID monitor element" on page 558

"client_platform - Client operating platform monitor element" on page 559

"client_port_number - Client port number monitor element" on page 560

"client_prdid - Client product and version ID monitor element" on page 560

"client_protocol - Client communication protocol monitor element" on page 561

"client_userid - Client user ID monitor element" on page 562

"client_wrkstnname - Client workstation name monitor element" on page 563

"connection_start_time - Connection start time monitor element" on page 579

"coord_partition_num - Coordinator partition number monitor element" on page 591

"destination_service_class_id – Destination service class ID monitor element" on page 618

"session_auth_id - Session authorization ID monitor element" on page 910

"source_service_class_id - Source service class ID monitor element" on page 917

"system_auth_id - System authorization identifier monitor element" on page 950

"threshold_action - Threshold action monitor element" on page 983

"threshold_maxvalue - Threshold maximum value monitor element" on page 984

"threshold_predicate - Threshold predicate monitor element" on page 984

"threshold_queuesize - Threshold queue size monitor element" on page 985

"thresholdid - Threshold ID monitor element" on page 985

"time_of_violation - Time of violation monitor element" on page 986

"uow_id - Unit of work ID monitor element" on page 1048

"workload_id - Workload ID monitor element" on page 1063

event_wlmetrics logical data group

Note: Monitor elements in this logical data group are calculated to show the change in value for each monitor element since the last time statistics were collected, or since database activation, which ever happed most recently.

"wlm_queue_time_total - Workload manager total queue time monitor element" on page 1060

"wlm_queue_assignments_total - Workload manager total queue assignments monitor element" on page 1059

"fcm_tq_recv_wait_time - FCM table queue received wait time monitor element" on page 656

"fcm_message_recv_wait_time - FCM message received wait time monitor element" on page 641

"fcm_tq_send_wait_time - FCM table queue send wait time monitor element" on page 659

"fcm_message_send_wait_time - FCM message send wait time monitor element" on page 645

"agent_wait_time - Agent wait time monitor element" on page 511

"agent_waits_total - Total agent waits monitor element" on page 512

"lock_wait_time - Time waited on locks monitor element" on page 734

"lock_waits - Lock waits monitor element" on page 737

"direct_read_time - Direct read time monitor element" on page 622

"direct_read_reqs - Direct read requests monitor element" on page 620

"direct_write_time - Direct write time monitor element" on page 627

"direct_write_reqs - Direct write requests monitor element" on page 626

"log_buffer_wait_time - Log buffer wait time monitor element" on page 740

"num_log_buffer_full - Number of times full log buffer caused agents to wait monitor element" on page 775

"log_disk_wait_time - Log disk wait time monitor element" on page 741

"log_disk_waits_total - Total log disk waits monitor element" on page 743

"tcpip_recv_wait_time - TCP/IP received wait time monitor element" on page 976

"tcpip_recvs_total - TCP/IP receives total monitor element" on page 977

"client_idle_wait_time - Client idle wait time monitor element" on page 557

"ipc_recv_wait_time - Interprocess communication received wait time monitor element" on page 705

"ipc_recvs_total - Interprocess communication receives total monitor element" on page 706

"ipc_send_wait_time - Interprocess communication send wait time monitor element" on page 708

"ipc_sends_total - Interprocess communication send total monitor element" on page 709

"tcpip_send_wait_time - TCP/IP send wait time monitor element" on page 979

"tcpip_sends_total - TCP/IP sends total monitor element" on page 980

"pool_write_time - Total buffer pool physical write time monitor element" on page 847

"pool_read_time - Total buffer pool physical read time monitor element" on page 833

"audit_file_write_wait_time - Audit file write wait time monitor element" on page 532

"audit_file_writes_total - Total audit files written monitor element" on page 533

"audit_subsystem_wait_time - Audit subsystem wait time monitor element" on page 534

"audit_subsystem_waits_total - Total audit subsystem waits monitor element" on page 536

"diaglog_write_wait_time - Diagnostic log file write wait time monitor element" on page 618

"diaglog_writes_total - Total diagnostic log file writes monitor element" on page 619

"fcm_send_wait_time - FCM send wait time monitor element" on page 652

"fcm_recv_wait_time - FCM received wait time monitor element" on page 648

"total_wait_time - Total wait time monitor element" on page 1038

"rqsts_completed_total - Total requests completed monitor element" on page 897

"total_rqst_time - Total request time monitor element" on page 1023

"app_rqsts_completed_total - Total application requests completed monitor element" on page 518

"total_app_rqst_time - Total application request time monitor element" on page 992

"total_section_sort_proc_time - Total section sort processing time monitor element" on page 1028

"total_section_sorts - Total section sorts monitor element" on page 1032

"total_section_sort_time - Total section sort time monitor element" on page 1030

"rows_read - Rows read monitor element" on page 892

"rows_modified - Rows modified monitor element" on page 890

"pool_data_l_reads - Buffer pool data logical reads monitor element" on page 816
"pool_index_l_reads - Buffer pool index logical reads monitor element" on page 825
"pool_temp_data_l_reads - Buffer pool temporary data logical reads monitor element" on page 835
"pool_temp_index_l_reads - Buffer pool temporary index logical reads monitor element" on page 839
"pool_xda_l_reads - Buffer pool XDA data logical reads monitor element" on page 849
"pool_temp_xda_l_reads - Buffer pool temporary XDA data logical reads monitor element" on page 843
"total_cpu_time - Total CPU time monitor element" on page 1001
"act_completed_total - Total completed activities monitor element" on page 495
"pool_data_p_reads - Buffer pool data physical reads monitor element" on page 818
"pool_temp_data_p_reads - Buffer pool temporary data physical reads monitor element" on page 837
"pool_xda_p_reads - Buffer pool XDA data physical reads monitor element" on page 851
"pool_temp_xda_p_reads - Buffer pool temporary XDA data physical reads monitor element" on page 845
"pool_index_p_reads - Buffer pool index physical reads monitor element" on page 827
"pool_temp_index_p_reads - Buffer pool temporary index physical reads monitor element" on page 841
"pool_data_writes - Buffer pool data writes monitor element" on page 820
"pool_xda_writes - Buffer pool XDA data writes monitor element" on page 853
"pool_index_writes - Buffer pool index writes monitor element" on page 829
"direct_reads - Direct reads from database monitor element" on page 624
"direct_writes - Direct writes to database monitor element" on page 629
"rows_returned - Rows returned monitor element" on page 893
"deadlocks - Deadlocks detected monitor element" on page 614
"lock_timeouts - Number of lock timeouts monitor element" on page 732
"lock_escals - Number of lock escalations monitor element" on page 721
"fcm_sends_total - FCM sends total monitor element" on page 654
"fcm_recvs_total - FCM receives total monitor element" on page 650
"fcm_send_volume - FCM send volume monitor element" on page 651
"fcm_recv_volume - FCM received volume monitor element" on page 647
"fcm_message_sends_total - Total FCM message sends monitor element" on page 646
"fcm_message_recvs_total - Total FCM message receives monitor element" on page 643
"fcm_message_send_volume - FCM message send volume monitor element" on page 644
"fcm_message_recv_volume - FCM message received volume monitor element" on page 640

"fcm_tq_sends_total - FCM table queue send total monitor element" on page 660

"fcm_tq_recvs_total - FCM table queue receives total monitor element" on page 657

"fcm_tq_send_volume - FCM table queue send volume monitor element" on page 658

"fcm_tq_recv_volume - FCM table queue received volume monitor element" on page 655

"tq_tot_send_spills - Total number of table queue buffers overflowed monitor element" on page 1044

"tcpip_send_volume - TCP/IP send volume monitor element" on page 978

"tcpip_recv_volume - TCP/IP received volume monitor element" on page 975

"ipc_send_volume - Interprocess communication send volume monitor element" on page 707

"ipc_recv_volume - Interprocess communication received volume monitor element" on page 704

"post_threshold_sorts - Post threshold sorts monitor element" on page 858

"post_shrthreshold_sorts - Post shared threshold sorts monitor element" on page 856

"sort_overflows - Sort overflows monitor element" on page 915

"audit_events_total - Total audit events monitor element" on page 531

"act_rejected_total - Total rejected activities monitor element" on page 497

"act_aborted_total - Total aborted activities monitor element" on page 494

"total_sorts - Total sorts monitor element" on page 1036

"total_routine_time - Total routine time monitor element" on page 1018

"total_compile_proc_time - Total compile processing time monitor element" on page 998

"total_compilations - Total compilations monitor element" on page 997

"total_compile_time - Total compile time monitor element" on page 999

"total_implicit_compile_proc_time - Total implicit compile processing time monitor element" on page 1004

"total_implicit_compilations - Total implicit complications monitor element" on page 1003

"total_implicit_compile_time - Total implicit compile time monitor element" on page 1005

"total_runstats_proc_time - Total runtime statistics processing time monitor element" on page 1025

"total_runstats - Total runtime statistics monitor element" on page 1024

"total_runstats_time - Total runtime statistics time monitor element" on page 1026

"total_reorg_proc_time - Total reorganization processing time monitor element" on page 1011

"total_reorgs - Total reorganizations monitor element" on page 1013

"total_reorg_time - Total reorganization time monitor element" on page 1012

"total_load_proc_time - Total load processing time monitor element" on page 1006

"total_loads - Total loads monitor element" on page 1008

"total_load_time - Total load time monitor element" on page 1007

"total_section_proc_time - Total section processing time monitor element" on page 1027

"total_app_section_executions - Total application section executions monitor element" on page 993

"total_section_time - Total section time monitor element" on page 1033

"total_commit_proc_time - Total commits processing time monitor element" on page 995

"total_app_commits - Total application commits monitor elements" on page 990

"total_commit_time - Total commit time monitor element" on page 996

"total_rollback_proc_time - Total rollback processing time monitor element" on page 1014

"total_app_rollbacks - Total application rollbacks monitor element" on page 991

"total_rollback_time - Total rollback time monitor element" on page 1015

"total_routine_user_code_proc_time - Total routine user code processing time monitor element" on page 1020

"total_routine_user_code_time - Total routine user code time monitor element" on page 1021

"thresh_violations - Number of threshold violations monitor element" on page 981

"num_lw_thresh_exceeded - Number of lock wait thresholds exceeded monitor element" on page 778

"total_routine_invocations - Total routine invocations monitor elements" on page 1016

"int_commits - Internal commits monitor element" on page 698

"int_rollbacks - Internal rollbacks monitor element" on page 700

"cat_cache_inserts - Catalog cache inserts monitor element" on page 547

"cat_cache_lookups - Catalog cache lookups monitor element" on page 548

"pkg_cache_inserts - Package cache inserts monitor element" on page 801

"pkg_cache_lookups - Package cache lookups monitor element" on page 802

"act_rqsts_total - Total activity requests monitor elements" on page 500

"total_act_wait_time - Total activity wait time monitor element" on page 989

"total_act_time - Total activity time monitor element" on page 988

FCM_TQ_RECV_WAITS_TOTAL

FCM_MESSAGE_RECV_WAITS_TOTAL

FCM_TQ_SEND_WAITS_TOTAL

FCM_MESSAGE_SEND_WAITS_TOTAL

FCM_SEND_WAITS_TOTAL

FCM_RECV_WAITS_TOTAL

event_wlstats logical data group

"act_cpu_time_top – Activity CPU time top monitor element" on page 496

"act_rows_read_top – Activity rows read top monitor element" on page 499

"concurrent_wlo_act_top - Concurrent WLO activity top monitor element" on page 569

"concurrent_wlo_top - Concurrent workload occurrences top monitor element" on page 569

"coord_act_aborted_total - Coordinator activities aborted total monitor element" on page 583

"coord_act_completed_total - Coordinator activities completed total monitor element" on page 584

"coord_act_est_cost_avg - Coordinator activity estimated cost average monitor element" on page 584

"coord_act_exec_time_avg - Coordinator activities execution time average monitor element" on page 585

"coord_act_interarrival_time_avg - Coordinator activity arrival time average monitor element" on page 586

"coord_act_lifetime_avg - Coordinator activity lifetime average monitor element" on page 587

"coord_act_lifetime_top - Coordinator activity lifetime top monitor element" on page 587

"coord_act_queue_time_avg - Coordinator activity queue time average monitor element" on page 588

"coord_act_rejected_total - Coordinator activities rejected total monitor element" on page 589

"cost_estimate_top - Cost estimate top monitor element" on page 592

details_xml*

"last_wlm_reset - Time of last reset monitor element" on page 714

"lock_wait_time_top – Lock wait time top monitor element" on page 736

metrics* (In Version 9.7 Fix Pack 6 and later fix packs, you can also access metrics reported in this document through the "event_wlmetrics logical data group" on page 484.)

"rows_returned_top - Actual rows returned top monitor element" on page 895

"statistics_timestamp - Statistics timestamp monitor element" on page 928

"uow_total_time_top - UOW total time top monitor element" on page 1052

"temp_tablespace_top - Temporary table space top monitor element" on page 980

"wlo_completed_total - Workload occurrences completed total monitor element" on page 1062

"workload_id - Workload ID monitor element" on page 1063

"workload_name - Workload name monitor element" on page 1064

*Two XML documents are included in the output of this logical data group. In the `metrics` document, the values of the elements show the change in value for the monitor element since the last time statistics were collected. The values of the elements contained in `details_xml` are not reset at each interval; they are reset only when the database is reactivated. These XML document are metrics documents of type system_metrics, as described in the XML schema document `sqllib/misc/DB2MonCommon.xsd`.

Important: Starting with Version 9.7 Fix Pack 6, the XML document details_xml is deprecated in the statistics event monitor, and might be removed in a future release. For more information, see Reporting of metrics in details_xml by the statistics event monitor has been deprecated"Reporting of metrics in details_xml by the statistics event monitor has been deprecated" in *What's New for DB2 Version 9.7*.

event_wcstats logical data group

"act_cpu_time_top – Activity CPU time top monitor element" on page 496

"act_rows_read_top – Activity rows read top monitor element" on page 499

"act_total - Activities total monitor element" on page 501
"coord_act_est_cost_avg - Coordinator activity estimated cost average monitor element" on page 584
"coord_act_exec_time_avg - Coordinator activities execution time average monitor element" on page 585
"coord_act_interarrival_time_avg - Coordinator activity arrival time average monitor element" on page 586
"coord_act_lifetime_avg - Coordinator activity lifetime average monitor element" on page 587
"coord_act_lifetime_top - Coordinator activity lifetime top monitor element" on page 587
"coord_act_queue_time_avg - Coordinator activity queue time average monitor element" on page 588
"cost_estimate_top - Cost estimate top monitor element" on page 592
"last_wlm_reset - Time of last reset monitor element" on page 714
"rows_returned_top - Actual rows returned top monitor element" on page 895
"statistics_timestamp - Statistics timestamp monitor element" on page 928
"temp_tablespace_top - Temporary table space top monitor element" on page 980
"work_action_set_id - Work action set ID monitor element" on page 1062
"work_action_set_name - Work action set name monitor element" on page 1062
"work_class_id - Work class ID monitor element" on page 1063
"work_class_name - Work class name monitor element" on page 1063

event_xact logical data group

"agent_id - Application handle (agent ID) monitor element" on page 506
"appl_id - Application ID monitor element" on page 519
"lock_escals - Number of lock escalations monitor element" on page 721
"lock_wait_time - Time waited on locks monitor element" on page 734
"locks_held_top - Maximum number of locks held monitor element" on page 739
"partial_record - Partial Record monitor element" on page 798
"prev_uow_stop_time - Previous Unit of Work Completion Timestamp" on page 861
"rows_read - Rows read monitor element" on page 892
"rows_written - Rows Written" on page 897
"sequence_no - Sequence number monitor element" on page 904
"system_cpu_time - System CPU time monitor element" on page 950
"tpmon_acc_str - TP monitor client accounting string monitor element" on page 1039
"tpmon_client_app - TP monitor client application name monitor element" on page 1040
"tpmon_client_userid - TP monitor client user ID monitor element" on page 1040
"tpmon_client_wkstn - TP monitor client workstation name monitor element" on page 1041
"uow_log_space_used - Unit of work log space used monitor element" on page 1049

"uow_start_time - Unit of work start timestamp monitor element" on page 1050
"uow_status - Unit of Work Status" on page 1051
"stop_time - Event Stop Time" on page 947
"user_cpu_time - User CPU time monitor element" on page 1053
"x_lock_escals - Exclusive lock escalations monitor element" on page 1066

lock logical data group

"data_partition_id - Data partition identifier monitor element" on page 604
"lock_attributes - Lock attributes monitor element" on page 717
"lock_count - Lock count monitor element" on page 718
"lock_current_mode - Original lock mode before conversion monitor element" on page 719
"lock_escalation - Lock escalation monitor element" on page 720
"lock_hold_count - Lock hold count monitor element" on page 723
"lock_mode - Lock mode monitor element" on page 724
"lock_name - Lock name monitor element" on page 726
"lock_object_name - Lock Object Name" on page 727
"lock_object_type - Lock object type waited on monitor element" on page 728
"lock_release_flags - Lock release flags monitor element" on page 730
"lock_status - Lock status monitor element" on page 730
"node_number - Node Number" on page 770
"table_file_id - Table file ID monitor element" on page 951
"table_name - Table name monitor element" on page 952
"table_schema - Table schema name monitor element" on page 953
"tablespace_name - Table space name monitor element" on page 961

sqlca logical data group

sqlcabc
sqlcaid
sqlcode
sqlerrd
sqlerrmc
sqlerrml
sqlerrp
sqlstate
sqlwarn

Logical data groups affected by COLLECT ACTIVITY DATA settings

The following table shows what logical data groups are collected when different COLLECT ACTIVITY DATA options are specified all types of WLM objects, including Service Subclass, Workload, Work Class (via a Work Action), and Threshold.

Table 63. COLLECT ACTIVITY DATA settings

Setting for COLLECT ACTIVITY DATA	Logical data groups collected
NONE	none

Table 63. COLLECT ACTIVITY DATA settings (continued)

Setting for COLLECT ACTIVITY DATA	Logical data groups collected
WITHOUT DETAILS	event_activity event_activitymetrics
WITH DETAILS	event_activity event_activitymetrics event_activitystmt
WITH DETAILS AND VALUES	event_activity event_activitymetrics event_activitystmt event_activityvals

Chapter 17. Database system monitor elements

A description of the data collected by the monitor element.

The monitor elements returned by the system monitor fall into the following categories:

- **Identification** for the database manager, an application, or a database connection being monitored.
- Data primarily intended to help you to **configure** the system.
- Database **activity** at various levels including database, application, table, or statement. This information can be used for activity monitoring, problem determination, and performance analysis. It can also be used for configuration.
- Information on **DB2 Connect** applications. Including information on DCS applications running at the gateway, SQL statements being executed, and database connections.
- Information on **Federated Database Systems**. This includes information about the total access to a data source by applications running in a DB2 federated system and information about access to a data source by a given application running in a federated server instance.

Monitor elements are described in a standard format as follows:

Element identifier
The name of the element. If parsing the data stream directly, the element identifier is uppercase and prefixed with SQLM_ELM_.

Element type
The type of information the monitor element returns. For example, the db2start_time monitor element returns a timestamp.

Snapshot monitoring information
If a monitor element returns snapshot monitoring information, a table with the following fields is shown.

- *Snapshot level*: The level of information that can be captured by the snapshot monitor. For example, the appl_status monitor element returns information at the Application level and at the Lock level.
- *Logical data grouping*: The logical data group where captured snapshot information is returned. If parsing the data stream directly, the logical data group identifier is uppercase and prefixed with SQLM_ELM_. For example, the appl_status monitor element returns information for the appl_id_info grouping and for the appl_lock_list grouping.
- *Monitor switch*: The system monitor switch that must be set to obtain this information. If the switch is Basic, data will always be collected for the monitor element.

Event monitoring information
If a monitor element is collected by event monitors, a table with the following fields is shown.

- *Event type*: The level of information that can be collected by the event monitor. The event monitor must be created with this event type to collect this information. For example, the appl_status monitor element is collected for CONNECTIONS event monitors.

© Copyright IBM Corp. 1993, 2012

- *Logical data grouping*: The logical data group where captured event information is returned. If parsing the data stream directly, the logical data group identifier is uppercase and prefixed with SQLM_ELM_. For example, the appl_status monitor element returns information for the event_conn grouping.
- *Monitor switch*: The system monitor switch that must be set to obtain this information. For event monitors, the TIMESTAMP switch is the only monitor switch that can restrict the collection of event data. If there is a dash shown for this field, data will always be collected for the monitor element.

Usage Information on how you can use the information collected by the monitor element when monitoring your database system.

acc_curs_blk - Accepted Block Cursor Requests

The number of times that a request for an I/O block was accepted.

Element identifier
acc_curs_blk

Element type
counter

Table 64. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

Table 65. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

Usage You can use this element in conjunction with *rej_curs_blk* to calculate the percentage of blocking requests that are accepted, rejected, or both.

See *rej_curs_blk* for suggestions on how to use this information to tune your configuration parameters.

act_aborted_total - Total aborted activities monitor element

The total number of coordinator activities at any nesting level that completed with errors. For service classes, if an activity is remapped to a different service subclass with a REMAP ACTIVITY action before it aborts, then this activity counts only towards the total of the subclass it aborts in.

Table 66. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 66. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 67. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element to understand if activities on the system are completing successfully. Activities may be aborted due to cancellation, errors, or reactive thresholds.

act_completed_total - Total completed activities monitor element

The total number of coordinator activities at any nesting level that completed successfully. For service classes, if an activity is remapped to a different subclass with a REMAP ACTIVITY action before it completes, then this activity counts only towards the total of the subclass it completes in.

Table 68. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE

Table 68. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 69. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element to determine the throughput of activities in the system.

act_cpu_time_top – Activity CPU time top monitor element

The high watermark for processor time used by activities at all nesting levels in a service class, workload, or work class. This value is reported in microseconds.

The monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service class or workload in which the activity runs is set to NONE. Activities contribute towards this high watermark only when request metrics are enabled. If activity metrics are not enabled, a value of 0 is returned.

For service classes, when you remap activities between service subclasses with a REMAP ACTIVITY action, only the act_cpu_time_top high watermark of the service subclass where an activity completes is updated, provided that a new high watermark is reached. The act_cpu_time_top high watermarks of other service subclasses an activity is mapped to but does not complete in are unaffected.

Table 70. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wcstats	-
Statistics	event_wlstats	-

Usage

Use this element to determine the highest amount of processor time used by an activity on a partition for a service class, workload, or work class during the time interval collected.

act_exec_time - Activity execution time monitor element

Time spent executing at this partition, in microseconds. For cursors, the execution time is the combined time for the open, the fetches, and the close. The time when the cursor is idle is not counted towards execution time. For routines, execution time is the start to end of routine invocation. The lifetimes of any cursors left open by routine (to return a result set) after the routine finishes are not counted towards the routine execution time. For all other activities, execution time is the difference between start time and stop time. In all cases, execution time does not include time spent initializing or queued.

Table 71. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

Usage

This element can be used alone to know the elapsed time spent executing the activity by DB2 on each partition. This element can also be used together with **time_started** and **time_completed** monitor elements on the coordinator partition to compute the idle time for cursor activities. You can use the following formula:

```
Cursor idle time = (time_completed - time_started) - act_exec_time
```

act_rejected_total - Total rejected activities monitor element

The total number of coordinator activities at any nesting level that were rejected instead of being allowed to execute.

Table 72. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE

Table 72. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 73. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element to help determine whether predictive thresholds and work actions that prevent execution are effective and whether they are too restrictive.

act_remapped_in – Activities remapped in monitor element

Count of the number of activities to be remapped into this service subclass since the last reset.

Table 74. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-

Usage

Use this count to determine whether the remapping of activities into the service subclass is occurring as desired.

act_remapped_out – Activities remapped out monitor element

Count of the number of activities to be remapped out of this service subclass since the last reset.

Table 75. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-

Usage

Use this count to determine whether the remapping of activities out of the service subclass is occurring as desired.

act_rows_read_top – Activity rows read top monitor element

The high watermark for the number of rows read by activities at all nesting levels in a service class, workload, or work class.

The monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service class or workload in which the activity runs is set to NONE. Activities contribute towards this high watermark only when request metrics are enabled. If activity metrics are not enabled, a value of 0 is returned.

For service classes, when you remap activities between service subclasses with a REMAP ACTIVITY action only the act_rows_read_top high watermark of the service subclass where an activity completes is updated, provided that a new high watermark is reached. The act_rows_read_top high watermarks of service subclasses an activity is mapped to but does not complete in are unaffected.

Table 76. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wcstats	-
Statistics	event_wlstats	-

Usage

Use this element to determine the highest number of rows read by an activity on a partition for a service class, workload, or work class during the time interval collected.

act_rqsts_total - Total activity requests monitor elements

The number of individual coordinator and subagent requests completed as part of an activity. For example, a fetch on a cursor activity.

Table 77. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 78. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

act_total - Activities total monitor element

Total number of activities at any nesting level that had work actions corresponding to the specified work class applied to them since the last reset.

Table 79. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wcstats	-

Usage

Every time an activity has one or more work actions associated with a work class applied to it, a counter for the work class is updated. This counter is exposed using the **act_total** monitor element. The counter can be used to judge the effectiveness of the work action set (for example, how many activities have a actions applied). It can also be used to understand the different types of activities on the system.

activate_timestamp - Activate timestamp monitor element

The time when an event monitor was activated.

Table 80. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activity	event_activity	-
Activity	event_activitystmt	-
Activity	event_activityvals	-
Threshold Violations	event_thresholdviolations	-

Usage

Use this element to correlate information returned by the above event types.

active_hash_joins - Active hash joins

The total number of hash joins that are currently running and consuming memory.

Table 81. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	-

active_olap_funcs - Active OLAP Functions monitor element

The total number of OLAP functions that are currently running and consuming sort heap memory.

Table 82. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	-

For snapshot monitoring, this counter can be reset.

active_sorts - Active Sorts

The number of sorts in the database that currently have a sort heap allocated.

Table 83. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Usage Use this value in conjunction with *sort_heap_allocated* to determine the average sort heap space used by each sort. If the *sortheap* configuration parameter is substantially larger than the average sort heap used, you may be able to lower the value of this parameter.

This value includes heaps for sorts of temporary tables that were created during relational operations.

activity_collected - Activity collected monitor element

This element indicates whether or not activity event monitor records are to be collected for a violated threshold.

Table 84. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Threshold violations	event_thresholdviolations	-

Usage

Use this element to determine whether to expect an activity event for the activity that violated the threshold to be written to the activity event monitor.

When an activity finishes or aborts and the activity event monitor is active at the time, if the value of this monitor element is 'Y', the activity that violated this threshold will be collected. If the value of this monitor element is 'N', it will not be collected.

activity_id - Activity ID monitor element

Counter which uniquely identifies an activity for an application within a given unit of work.

Table 85. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 86. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Activities	event_activity	-

Table 86. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activitystmt	-
Activities	event_activityvals	-
Threshold violations	event_thresholdviolations	-

Usage

Use this element in conjunction with other activity history elements for analysis of the behavior of an activity.

To uniquely identify an activity outside its unit of work, use the combination of **activity_id** and **uow_id** plus one of the following: **appl_id** or **agent_id**.

activity_secondary_id - Activity secondary ID monitor element

The value for this element is incremented each time an activity record is written for the same activity. For example, if an activity record is written once as a result of having called the WLM_CAPTURE_ACTIVITY_IN_PROGRESS procedure and a second time when the activity ends, the element would have a value of 0 for the first record and 1 for the second record.

Table 87. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-
Activities	event_activitystmt	-
Activities	event_activityvals	-

Usage

Use this element with **activity_id**, **uow_id**, and **appl_id** monitor elements to uniquely identify activity records when information about the same activity has been written to the activities event monitor multiple times.

For example, information about an activity would be sent to the activities event monitor twice in the following case:

- the WLM_CAPTURE_ACTIVITY_IN_PROGRESS stored procedure was used to capture information about the activity while it was running
- information about the activity was collected when the activity completed, because the COLLECT ACTIVITY DATA clause was specified on the service class with which the activity is associated

activity_state - Activity state monitor element

The current state of the activity.

Table 88. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this monitor element to understand what the activity is currently doing (for example, is the activity stuck in a queue or waiting for input from the client). Possible values include:

- CANCEL_PENDING
- EXECUTING
- IDLE
- INITIALIZING
- QP_CANCEL_PENDING
- QP_QUEUED
- QUEUED
- TERMINATING
- UNKNOWN

activity_type - Activity type monitor element

The type of the activity.

Table 89. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 90. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

Usage

The possible values are:

- LOAD
- READ_DML
- WRITE_DML
- DDL
- CALL

- OTHER

The value OTHER is returned for SET statements that do not perform SQL (for example, SET special register, or SET EVENT MONITOR STATE) and the LOCK TABLE statement.

activitytotaltime_threshold_id - Activity total time threshold ID monitor element

The ID of the ACTIVITYTOTALTIME threshold that was applied to the activity.

Table 91. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand which ACTIVITYTOTALTIME threshold, if any, was applied to the activity.

activitytotaltime_threshold_value - Activity total time threshold value monitor element

A timestamp that is computed by adding the ACTIVITYTOTALTIME threshold duration to the activity entry time. If the activity is still executing when this timestamp is reached, the threshold will be violated.

Table 92. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand the value of the ACTIVITYTOTALTIME threshold applied to the activity, if any.

activitytotaltime_threshold_violated - Activity total time threshold violated monitor element

This monitor element returns 'Yes' to indicate that the activity violated the ACTIVITYTOTALTIME threshold. 'No' indicates that the activity has not yet violated the threshold.

Table 93. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to determine if the activity violated the ACTIVITYTOTALTIME threshold that was applied to the activity.

adapter_name - Adapter name monitor element

Name of the network adapter on this host.

Table 94. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_NETWORK_RESOURCES table function - Return network adapter information	Always collected

address - IP address from which the connection was initiated

The IP address from which the activity connection was initiated.

Table 95. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

Usage

Use this to identify the IP address from which the activity connection was initiated. Secure domain names are shown converted to an IP address.

agent_id - Application handle (agent ID) monitor element

A system-wide unique ID for the application. On a single-partitioned database, this identifier consists of a 16–bit counter. On a multi-partitioned database, this identifier consists of the coordinating partition number concatenated with a 16–bit counter. In addition, this identifier is the same on every partition where the application may make a secondary connection.

Table 96. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	Always collected
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected
MON_GET_LOCKS table function - list all locks in the currently connected database	Always collected
MON_GET_MEMORY_POOL table function - Get memory pool information	Always collected

Table 97. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_id_info	Basic
Lock	appl_lock_list	Basic
DCS Application	dcs_appl_info	Basic
Transaction	event_xact	-

Table 98. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-
Connections	event_connheader	-
Statements	event_stmt	-
Statements	event_subsection	-
Deadlocks[1]	event_dlconn	-
Deadlocks with Details[1]	event_detailed_dlconn	-
Threshold violations	event_thresholdviolations	-
Activities	event_activity	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

The application handle, also known as the agent ID, can be used to uniquely identify an active application.

Note: The **agent_id** monitor element has different behavior depending on your version of DB2. When taking snapshots from DB2 with version SQLM_DBMON_VERSION1 or SQLM_DBMON_VERSION2 to a DB2 (Version 5 or greater) database, the **agent_id** returned is not usable as an application identifier, rather it is the **agent_pid** of the agent serving the application. In these cases an **agent_id** is still returned for compatibility with earlier releases, but internally the DB2 database server will not recognize the value as an **agent_id**.

This value can be used as input to GET SNAPSHOT commands that require an agent ID or to the monitor table functions that require an application handle.

When reading event traces, it can be used to match event records with a given application.

It can also be used as input to the FORCE APPLICATION command or API. On multi-node systems this command can be issued from any node where the application has a connection. Its effect is global.

agent_id_holding_lock - Agent ID Holding Lock

The application handle of the agent holding a lock for which this application is waiting. The lock monitor group must be turned on to obtain this information.

Table 99. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Lock
Lock	appl_lock_list	Lock
Lock	lock_wait	Lock

Usage This element can help you determine which applications are in contention for resources.

If this element is 0 (zero) and the application is waiting for a lock, this indicates that the lock is held by an indoubt transaction. You can use either appl_id_holding_lk or the command line processor LIST INDOUBT TRANSACTIONS command (which displays the application ID of the CICS agent that was processing the transaction when it became indoubt) to determine the indoubt transaction, and then either commit it or roll it back.

Note that more than one application can hold a shared lock on an object for which this application is waiting. See lock_mode for information about the type of lock that the application holds. If you are taking an application snapshot, only one of the agent IDs holding a lock on the object will be returned. If you are taking a lock snapshot, all of the agent IDs holding a lock on the object will be identified.

agent_pid - Engine dispatchable unit (EDU) identifier monitor element

The unique identifier for the engine dispatchable unit (EDU) for the agent. Except on the Linux operating system, the EDU ID is mapped to the thread ID. On the Linux operating system, the EDU ID is a DB2 generated unique identifier.

Table 100. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	agent	Statement

Table 101. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-

Usage

You can use this element to link database system monitor information to other sources of diagnostic information, such as system traces. You can also use it to monitor how agents working for a database application use system resources.

agent_status - DCS Application Agents

In a connection concentrator environment, this value shows which applications currently have associated agents.

Element identifier
agent_status

Element type
information

Table 102. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcs_appl_info	Basic

Usage Values are:

- SQLM_AGENT_ASSOCIATED

  The agent working on behalf of this application is associated with it.
- SQLM_AGENT_NOT_ASSOCIATED

  The agent that was working on behalf of this application is no longer associated with it and is being used by another application. The next time work is done for this application without an associated agent, an agent will be re-associated.

agent_sys_cpu_time - System CPU Time used by Agent

The total *system* CPU time (in seconds and microseconds) used by the database manager agent process.

Element identifier
agent_sys_cpu_time

Element type
time

Table 103. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Timestamp

For snapshot monitoring at the application level, this counter can be reset. This counter cannot be reset at other levels.

Usage This element along with the other related CPU-time elements can help you understand the level of activity within an application, and may help you identify applications that could benefit from additional tuning.

This element includes CPU time for both SQL and non-SQL statements, as well as CPU time for any unfenced user-defined functions (UDFs)

System CPU represents the time spent in system calls. User CPU represents time spent executing database manager code.

Note: If this information is not available for your operating system, this element will be set to 0.

agent_usr_cpu_time - User CPU Time used by Agent

The total CPU time (in seconds and microseconds) used by database manager agent process.

Element identifier
agent_usr_cpu_time

Element type
time

Table 104. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Timestamp

For snapshot monitoring, this counter can be reset.

Usage This element along with the other CPU-time related elements can help you identify applications or queries that consume large amounts of CPU.

This counter includes time spent on both SQL and non-SQL statements, as well as any unfenced user-defined functions (UDFs) or stored procedures executed by the application.

System CPU represents the time spent in system calls. User CPU represents time spent executing database manager code.

Note: If this information is not available for your operating system, this element will be returned as 0.

agent_wait_time - Agent wait time monitor element

Time spent by an application queued to wait for an agent under concentrator configurations. The value is given in milliseconds.

Table 105. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 106. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

The **agent_wait_time** monitor element can be used to help evaluate how efficiently your system is running in a concentrator environment. A high agent wait relative to the **total_request_time** monitor element value indicates that requests are spending a lot of time queued waiting for agents, which may be indicative of one or more of the following:

- The **max_coordagents** configuration parameter has been configured too small for your workload. You may need to increase the value of **max_coordagents** configuration parameter, or the ratio of **max_coordagents** configuration parameter to **max_connections** configuration parameter if you are running with both parameters set to AUTOMATIC, to ensure that enough coordinator agents are available to service your application requests in a timely manner.
- Your workload is not committing frequently enough. For the concentrator to work efficiently, applications should issue commits relatively frequently to ensure that their agents can be freed up to serve requests on other applications. If your applications do not do frequent commits you may need to configure a proportionally higher number of coordinator agents to reduce the time spent waiting for agents to become available.

agent_waits_total - Total agent waits monitor element

Number of times an application had to wait for an agent to be assigned under concentrator configurations.

Table 107. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 108. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Statistics	event_scmetrics*	REQUEST METRICS BASE

Table 108. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element in conjunction with the **agent_wait_time** monitor element to determine the average amount of time an application request spends waiting for an agent in a concentrator environment.

agents_created_empty_pool - Agents Created Due to Empty Agent Pool

The number of agents created because the agent pool was empty. It includes the number of agents started at DB2 start up (*num_initagents)*.

Table 109. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

Usage In conjunction with agents_from_pool, you can calculate the ratio of

```
Agents Created Due to Empty Agent Pool / Agents Assigned From Pool
```

See agents_from_pool for information on using this element.

agents_from_pool - Agents Assigned From Pool

The number of agents assigned from the agent pool.

Table 110. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

Usage

This element can be used with the **agents_created_empty_pool** monitor element to determine how often an agent must be created because the pool is empty.

The following ratio

Agents Created Due to Empty Agent Pool / Agents Assigned From Pool

can be used to help set an appropriate value for the **num_poolagents** configuration parameter.

For most users, the default value of 100 with AUTOMATIC will ensure optimal performance.

This ratio may fluctuate somewhat with the workload. At times of low activity on the system, additional agent creation and termination may occur. At times of high activity on the system, more agent reuse will occur. A low ratio indicates that there is a high amount of agent reuse, which is expected on systems with high activity. A high ratio indicates a higher amount of agent creation than reuse is occurring. If this is a concern, increase the value for the **num_poolagents** configuration parameter to lower the ratio. However, this will cause additional resources consumption on the system.

agents_registered - Agents Registered

The number of agents registered in the database manager instance that is being monitored (coordinator agents and subagents).

Table 111. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

Usage

Use this element to help evaluate your settings for the **max_coordagents** and **max_connections** configuration parameters, as well as the intraquery parallelism settings.

agents_registered_top - Maximum Number of Agents Registered

The maximum number of agents that the database manager has ever registered, at the same time, since it was started (coordinator agents and subagents).

Table 112. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

Usage

You may use this element to help you evaluate your settings for the **max_coordagents** and **max_connections** configuration parameters, as well as the intraquery parallelism settings.

The number of agents registered at the time the snapshot was taken is recorded by the agents_registered monitor element.

agents_stolen - Stolen Agents

At the database manager snapshot level, this monitor element represents the number of idle agents associated with an application which get reassigned to work on a different application. At the application snapshot level, this monitor element represents the number of idle agents associated with a different application which get reassigned to work on this application.

Table 113. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

Table 113. Snapshot Monitoring Information (continued)

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Usage

The **num_poolagents** configuration parameter is set to AUTOMATIC by default. This means that DB2 automatically manages the pooling of idle agents, which includes assigning work to idle agents associated with another application.

agents_top - Number of Agents Created

At the application level, this is the maximum number of agents that were used when executing the statement. At the database level, it is the maximum number of agents for all applications.

Element identifier
agents_top

Element type
watermark

Table 114. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Statement
Application	stmt	Statement

Usage An indicator how well intra-query parallelism was realized.

agents_waiting_on_token - Agents Waiting for a Token

The number of agents waiting for a token so they can execute a transaction in the database manager.

Note: The **agents_waiting_on_token** monitor element is deprecated starting with DB2 Version 9.5. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

Table 115. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

Usage

You can use this element to help evaluate your setting for the **maxcagents** configuration parameter.

Each application has a dedicated coordinator agent to process database requests within the database manager. Each agent has to get a token before it can execute a transaction. The maximum number of agents that can execute database manager

transactions is limited by the configuration parameter **maxcagents**.

agents_waiting_top - Maximum Number of Agents Waiting monitor element

The maximum number of agents that have ever been waiting for a token, at the same time, since the database manager was started.

Note: The **agents_waiting_top** monitor element is deprecated starting with DB2 Version 9.5. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

Table 116. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

Usage

Use this element to help you evaluate your setting of the **maxcagents** configuration parameter.

The number of agents waiting for a token at the time the snapshot was taken is recorded by the **agents_waiting_on_token** monitor element.

If the **maxcagents** parameter is set to its default value (-1), no agents should wait for a token and the value of this monitor element should be zero.

agg_temp_tablespace_top - Aggregate temporary table space top monitor element

The high watermark in KB for the aggregate temporary table space usage of DML activities at all nesting levels in a service class. The aggregate is computed by summing the temporary table space usage across all activities in the service subclass, and this high watermark represents the highest value reached by this aggregate since the last reset. The monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service class is set to NONE. An AGGSQLTEMPSPACE threshold must be defined and enabled for at least one service subclass in the same superclass as the subclass to which this record belongs, otherwise a value of 0 is returned.

Table 117. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-

Usage

Use this element to determine the highest aggregate DML activity system temporary table space usage reached on a partition for a service subclass in the time interval collected.

aggsqltempspace_threshold_id - Aggregate SQL temporary space threshold ID monitor element

The numeric ID of the AGGSQLTEMPSPACE threshold that was applied to the activity.

Table 118. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand which AGGSQLTEMPSPACE threshold, if any, was applied to the activity.

aggsqltempspace_threshold_value - AggSQL temporary space threshold value monitor element

The upper bound of the AGGSQLTEMPSPACE threshold that was applied to the activity.

Table 119. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand the value of the AGGSQLTEMPSPACE threshold applied to the activity, if any.

aggsqltempspace_threshold_violated - AggSQL temporary space threshold violated monitor element

The optional monitor element when set to ‘Yes' indicates that the activity violated the AGGSQLTEMPSPACE threshold that was applied to it. ‘No' indicates that the activity has not yet violated the threshold.

Table 120. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to determine if the activity violated the AGGSQLTEMPSPACE threshold that was applied to the activity.

app_rqsts_completed_total - Total application requests completed monitor element

Total number of external (application) requests executed by the coordinator. For service subclasses, this monitor element is updated only for the subclass where the application request completes.

Table 121. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 122. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this monitor element to understand how many requests are being submitted into the system from applications.

appl_con_time - Connection Request Start Timestamp

The date and time that an application started a connection request.

Element identifier
appl_con_time

Element type
timestamp

Table 123. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Timestamp

Usage Use this element to determine when the application started its connection request to the database.

appl_id - Application ID monitor element

This identifier is generated when the application connects to the database at the database manager or when DB2 Connect receives a request to connect to a DRDA® database.

Table 124. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	Always collected
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected

Table 125. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_id_info	Basic
DCS Application	dcs_appl_info	Basic
Lock	appl_lock_list	Basic

Table 126. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-
Connection	event_conn	-
Connections	event_connheader	-
Statements	event_stmt	-
Transactions[1]	event_xact	-

Table 126. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks[2]	event_dlconn	-
Deadlocks with Details[2]	event_detailed_dlconn	-
Activities	event_activitystmt	-
Activities	event_activity	-
Activities	event_activityvals	-
Threshold violations	event_thresholdviolations	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR UNIT OF WORK statement to monitor transaction events.

2 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

This ID is known on both the client and server, so you can use it to correlate the client and server parts of the application. For DB2 Connect applications, you will also need to use **outbound_appl_id** monitor element to correlate the client and server parts of the application.

This identifier is unique across the network. There are different formats for the application ID, which are dependent on the communication protocol between the client and the server machine on which the database manager, DB2 Connect, or both are running. Each of the formats consists of three parts separated by periods.

1. TCP/IP

   Format

   `IPAddr.Port.Timestamp`

   IPv4

   Example

   `9.26.120.63.43538.090924175700`

   Details

   In IPv4, a TCP/IP-generated application ID is composed of three sections. The first section is the IP address. It is represented as four decimal numbers of the form `a.b.c.d`. The second section is the port number, which is represented as 5 decimal characters. The third section is the approximate timestamp, represented as 12 decimal characters.

   IPv6

   Example

   `2002:91a:519:13:20d:60ff:feef:cc64.5309.090924175700`

   Details

   In IPv6, a TCP/IP-generated application ID is composed of three sections. The first section contains the IPv6 address of the form a:b:c:d:e:f:g:h, where each of a-h is up to 4 hexadecimal

digits. The second section is the port number. The third section is the approximate timestamp identifier for the instance of this application.

2. Local Applications

   Format

   ```
   *LOCAL.DB2 instance.Application instance
   ```

   Example

   ```
   *LOCAL.DB2INST1.930131235945
   ```

   Details

   The application ID generated for a local application is made up by concatenating the string `*LOCAL`, the name of the DB2 instance, and a unique identifier for the instance of this application.

   For multiple database partition instances, `LOCAL` is replaced with `Nx`, where `x` is the partition number from which the client connected to the database. For example, `*N2.DB2INST1.0B5A12222841`.

Use the **client_protocol** monitor element to determine which communications protocol the connection is using and, as a result, the format of the **appl_id** monitor element.

appl_id_holding_lk - Application ID Holding Lock

The application ID of the application that is holding a lock on the object that this application is waiting to obtain.

Element identifier
appl_id_holding_lk

Element type
information

Table 127. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Lock
Lock	appl_lock_list	Lock
Lock	lock_wait	Lock

Table 128. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	event_dlconn	-
Deadlocks with Details	event_detailed_dlconn	-

Usage This element can help you determine which applications are in contention for resources. Specifically, it can help you identify the application handle (agent ID) and table ID that are holding the lock. Note that you may use the LIST APPLICATIONS command to obtain information to relate the application ID with an agent ID. However, it is a good idea to collect this type of information when you take the snapshot, as it could be unavailable if the application ends before you run the LIST APPLICATIONS command.

Note that more than one application can hold a shared lock on an object for which this application is waiting to obtain a lock. See lock_mode for

information about the type of lock that the application holds. If you are taking an application snapshot, only one of the application IDs holding a lock on the object will be returned. If you are taking a lock snapshot, all of the application IDs holding a lock on the object will be returned.

appl_id_oldest_xact - Application with Oldest Transaction

The application ID (which corresponds to the *agent_id* value from the application snapshot) of the application that has the oldest transaction.

Element identifier
appl_id_oldest_xact

Element type
information

Table 129. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Usage This element can help you determine which application has the oldest active transaction. This application can be forced to free up log space. If it is taking up a great deal of log space, you should examine the application to determine if it can be modified to commit more frequently.

There are times when there is not a transaction holding up logging, or the oldest transaction does not have an application ID (for example, indoubt transaction or inactive transaction). In these cases, this application's ID is not returned in the data stream.

appl_idle_time - Application Idle Time

Number of seconds since an application has issued any requests to the server. This includes applications that have not terminated a transaction, for example not issued a commit or rollback.

Element identifier
appl_idle_time

Element type
information

Table 130. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Statement
DCS Application	dcs_appl	Statement

Usage This information can be used to implement applications that force users that have been idle for a specified number of seconds.

appl_name - Application name monitor element

The name of the application running at the client, as known to the database or DB2 Connect server.

Table 131. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	Always collected
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected

Table 132. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_id_info	Basic
Lock	appl_lock_list	Basic
DCS Application	dcs_appl_info	Basic

Table 133. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-
Connections	event_connheader	-
Activities	event_activity	-
Threshold violations	event_thresholdviolations	-

Usage

This element can be used with **appl_id** to relate data items with your application.

In a client-server environment, this name is passed from the client to the server when establishing the database connection. Any non-English characters in the application name will be removed. A CLI application can set the SQL_ATTR_INFO_PROGRAMNAME attribute with a call to SQLSetConnectAttr. When SQL_ATTR_INFO_PROGRAMNAME is set before the connection to the server is established, the value specified overrides the actual client application name and will be the value that is displayed in the **appl_name** monitor element.

In situations where the client application code page is different from the code page under which the database system monitor is running, you can use **codepage_id** to help translate **appl_name**.

appl_priority - Application Agent Priority

The priority of the agents working for this application.

Element identifier
appl_priority

Element type
information

Table 134. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

Table 135. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

Usage You can use this element to check if applications are running with the expected priorities. Application priorities can be set by an administrator. They can be changed by the governor utility (**db2gov**).

The governor is used by DB2 to monitor and change the behavior of applications running against a database. This information is used to schedule applications and balance system resources.

A governor daemon collects statistics about the applications by taking snapshots. It checks these statistics against the rules governing applications running on that database. If the governor detects a rule violation, it takes the appropriate action. These rules and actions were specified by you in the governor configuration file.

If the action associated with a rule is to change an application's priority, the governor changes the priority of the agents in the partition where the violation was detected.

appl_priority_type - Application Priority Type

Operating system priority type for the agent working on behalf of the application.

Element identifier
appl_priority_type

Element type
information

Table 136. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

Table 137. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

Usage Dynamic priority is recalculated by the operating system based on usage. Static priority does not change.

appl_section_inserts - Section Inserts monitor element

Inserts of SQL sections by an application from its shared SQL workspace.

Table 138. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

Table 139. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage

The working copy of any executable section is stored in a shared SQL workspace. This is a count of when a copy was not available and had to be inserted.

appl_section_lookups - Section Lookups

Lookups of SQL sections by an application from its shared SQL workspace.

Table 140. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 141. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage

Each agent has access to a shared SQL workspace where the working copy of any executable section is kept. This counter indicates how many times the SQL work area was accessed by agents for an application.

appl_status - Application Status

The current status of the application.

Table 142. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_id_info	Basic
Lock	appl_lock_list	Basic

Table 143. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Connection	event_conn	-

Usage

This element can help you diagnose potential application problems. Values for this field are listed in the following table.

API Constant	Description
SQLM_AUTONOMOUS_WAIT	**Autonomous Wait:** The application is waiting for an autonomous routine to complete.
SQLM_BACKUP	**Backing Up Database:** The application is performing a backup of the database.
SQLM_COMMIT_ACT	**Commit Active:** The unit of work is committing its database changes.
SQLM_COMP	**Compiling:** The database manager is compiling an SQL statement or precompiling a plan on behalf of the application.
SQLM_CONNECTED	**Database Connect Completed:** The application has initiated a database connection and the request has completed.
SQLM_CONNECTPEND	**Database Connect Pending:** The application has initiated a database connection but the request has not yet completed.
SQLM_CREATE_DB	**Creating Database:** The agent has initiated a request to create a database and that request has not yet completed.
SQLM_DECOUPLED	**Decoupled from Agent:** There are no agents currently associated with the application. This is a normal state. When the Connection Concentrator is enabled, there is no dedicated coordinator agent, so an application can be decoupled on the coordinator partition. In non-concentrator environments, an application cannot be decoupled on the coordinator partition as there will always be a dedicated coordinator agent .
SQLM_DISCONNECTPEND	**Database Disconnect Pending:** The application has initiated a database disconnect but the command has not yet completed executing. The application may not have explicitly executed the database disconnect command. The database manager will disconnect from a database if the application ends without disconnecting.
SQLM_INTR	**Request Interrupted:** An interrupt of a request is in progress.

API Constant	Description
SQLM_IOERROR_WAIT	**Wait to Disable Table space:** The application has detected an I/O error and is attempting to disable a particular table space. The application has to wait for all other active transactions on the table space to complete before it can disable the table space.
SQLM_LOAD	**Data Fast Load:** The application is performing a "fast load" of data into the database.
SQLM_LOCKWAIT	**Lock Wait:** The unit of work is waiting for a lock. After the lock is granted, the status is restored to its previous value.
SQLM_QUIESCE_TABLESPACE	**Quiescing a Table space:** The application is performing a quiesce table space request.
SQLM_RECOMP	**Recompiling:** The database manager is recompiling (that is, rebinding) a plan on behalf of the application.
SQLM_REMOTE_RQST	**Federated request pending:** The application is waiting for results from a federated data source.
SQLM_RESTART	**Restarting Database:** The application is restarting a database in order to perform crash recovery.
SQLM_RESTORE	**Restoring Database:** The application is restoring a backup image to the database.
SQLM_ROLLBACK_ACT	**Rollback Active:** The unit of work is rolling back its database changes.
SQLM_ROLLBACK_TO_SAVEPOINT	**Rollback to savepoint:** The application is rolling back to a savepoint.
SQLM_TEND	**Transaction Ended:** The unit of work is part of a global transaction that has ended but has not yet entered the prepared phase of the two-phase commit protocol.
SQLM_THABRT	**Transaction Heuristically Rolled Back:** The unit of work is part of a global transaction that has been heuristically rolled-back.
SQLM_THCOMT	**Transaction Heuristically Committed:** The unit of work is part of a global transaction that has been heuristically committed.
SQLM_TPREP	**Transaction Prepared:** The unit of work is part of a global transaction that has entered the prepared phase of the two-phase commit protocol.
SQLM_UNLOAD	**Data Fast Unload:** The application is performing a "fast unload" of data from the database.
SQLM_UOWEXEC	**Unit of Work Executing:** The database manager is executing requests on behalf of the unit of work.
SQLM_UOWQUEUED	**Unit of Work queued (Version 9.7 Fix Pack 1 and later):** The unit of work is queued, waiting for another activity to complete execution. The unit of work is queued because the threshold for the number of concurrently executing activities has been reached. For more information, see "Activity queuing" in the *Workload Manager Guide and Reference*.

API Constant	Description
SQLM_UOWWAIT	**Unit of Work waiting:** The database manager is waiting on behalf of the unit of work in the application. This status typically means that the system is executing in the application's code.
SQLM_WAITFOR_REMOTE	**Pending remote request:** The application is waiting for a response from a remote partition in a partitioned database instance.

application_handle - Application handle monitor element

A system-wide unique ID for the application. On a single-partitioned database, this identifier consists of a 16–bit counter. On a multi-partitioned database, this identifier consists of the coordinating partition number concatenated with a 16–bit counter. In addition, this identifier is the same on every partition where the application may make a secondary connection.

Table 144. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	Always collected
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected
MON_GET_LOCKS table function - list all locks in the currently connected database	Always collected
MON_GET_MEMORY_POOL table function - Get memory pool information	Always collected

Table 145. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_id_info	Basic
Lock	appl_lock_list	Basic
DCS Application	dcs_appl_info	Basic
Transaction	event_xact	-

Table 146. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-

Table 146. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Connections	event_connheader	-
Statements	event_stmt	-
Statements	event_subsection	-
Deadlocks[1]	event_dlconn	-
Deadlocks with Details[1]	event_detailed_dlconn	-
Threshold violations	event_thresholdviolations	-
Activities	event_activity	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

This monitor element is an alias of the **agent_id** monitor element.

When returned by MON_GET_MEMORY_POOL, this monitor element is NULL except when the memory pool being described is one of the following:

- APPLICATION
- STATISTICS
- STATEMENT
- SORT_PRIVATE.

appls_cur_cons - Applications Connected Currently

Indicates the number of applications that are currently connected to the database.

Table 147. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Lock	db_lock_list	Basic

Usage You may use this element to help you understand the level of activity within a database and the amount of system resource being used.

It can help you adjust the setting of the *maxappls* and *max_coordagents* configuration parameters. For example, its value is always the same as *maxappls*, you may want to increase the value of *maxappls*. See the *rem_cons_in* and the *local_cons* monitor elements for more information.

appls_in_db2 - Applications Executing in the Database Currently

Indicates the number of applications that are currently connected to the database, and for which the database manager is currently processing a request.

Table 148. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

arm_correlator - Application response measurement correlator monitor element

Identifier of a transaction in the Application Response Measurement (ARM) standard.

Table 149. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

Usage

This element can be used to link an activity collected by the activities event monitor to the applications associated with the activity, if such applications also support the Application Response Measurement (ARM) standard.

associated_agents_top - Maximum Number of Associated Agents

The maximum number of subagents associated with this application.

Table 150. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

async_runstats – Total number of asynchronous RUNSTATS requests monitor element

The total number of successful asynchronous RUNSTATS activities performed by real-time statistics gathering for all the applications in the database. Values reported by all the database partitions are aggregated together.

Table 151. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Statement

For snapshot monitoring, this counter can be reset.

Table 152. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage

Use this element to determine how many successful asynchronous RUNSTATS activities have been performed by real-time statistics gathering. This value changes frequently. In order to get a better view of the system usage, take a snapshot at specific intervals over an extended period of time. When used in conjunction with **sync_runstats** and **stats_fabrications** monitor elements, this element can help

you to track the different types of statistics collection activities related to real-time statistics gathering and analyze their performance impact.

audit_events_total - Total audit events monitor element

The total number of audit events generated.

Table 153. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 154. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

audit_file_write_wait_time - Audit file write wait time monitor element

Time spent waiting to write an audit record. The value is given in milliseconds.

Table 155. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 156. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this monitor element to determine the amount of time an agent spends waiting to open and write an audit event synchronously to disk.

In a typical scenario, only one agent attempts to open the audit log file at a time, as the other agents wait for access to the audit common subsystem before opening the file. Therefore, the wait time usually represents the time spent waiting to write the file to disk by the operating system. Audit utilities might lock the audit log file during execution, which causes a longer than normal wait time for agents to open and write to the audit log file. If asynchronous auditing is enabled, audit events that are larger than the asynchronous audit buffer are written directly to disk, instead of to the buffer, and contribute to the wait time.

Outside of the special audit utility scenario, the wait time depends on the speed of the disks and how quickly the operating system can write the data to them. In order to reduce this wait time for a given application and audit configuration, you might tune the operating system or use faster disks.

audit_file_writes_total - Total audit files written monitor element

The total number of times an agent has had to wait to write an audit event directly to disk.

Table 157. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 158. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE

Table 158. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this monitor element in conjunction with the **audit_file_write_wait_time** monitor element to determine the average time an application request spends waiting to open and write an audit event synchronously to disk.

audit_subsystem_wait_time - Audit subsystem wait time monitor element

Time spent waiting for space in audit buffer. Waiting occurs when audit buffer is full and agent must wait for audit daemon to write buffer to disk. The value is given in milliseconds.

Table 159. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 160. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE

Table 160. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this monitor element to determine the amount of time an agent spends waiting to access the audit common subsystem, while the audit common subsystem is busy handling events for other agents.

Certain common portions of the audit subsystem can only be accessed by a single agent at a time. The value of this monitor element indicates the amount of time that an agent must wait to access the audit common subsystem. This includes time spent by an agent that has filled the current asynchronous buffer waiting for the audit daemon to finish writing out a previous asynchronous buffer to disk. Other agents that are waiting while writing to the audit log file or waiting to make a request of the audit daemon have also accessed the audit common subsystem and wait times there will be reflected in this value.

To reduce this wait time, you might change the value of the **audit_buf_sz** configuration parameter if asynchronous auditing is in use. You can increase the value of the **audit_buf_sz** configuration parameter until further increases no longer show any reductions in the audit common subsystem wait time. At this point, the asynchronous buffers are large enough such that the daemon is able to write one full buffer to disk before the next buffer is full, and then the daemon is no longer a bottleneck. If the value of the **audit_buf_sz** configuration parameter must be increased to such an extent that too many audit records could be lost if a system failure were to occur, then you might reduce the wait time by tuning the operating system or using faster disks. If further reduction in the wait time is necessary, then use audit policies to reduce the number of audit events generated.

audit_subsystem_waits_total - Total audit subsystem waits monitor element

Number of times audit has waited for a buffer write.

Table 161. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 162. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this monitor element to determine the total number of times an agent has had to access the audit common subsystem. The generation of one audit event may

need to access the audit common subsystem none, one, or more times to record the event. Use the **audit_events_total** monitor element to determine the exact number of audit events generated.

auth_id - Authorization ID

The authorization ID of the user who invoked the application that is being monitored. On a DB2 Connect gateway node, this is the user's authorization ID on the host.

Table 163. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_id_info	Basic
Lock	appl_lock_list	Basic
DCS Application	dcs_appl_info	Basic

Table 164. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-
Connections	event_connheader	-

Usage

In an explicit trusted connection, the **auth_id** value does not change immediately when you switch users. Rather, the **auth_id** is updated the first time you access the database after switching users. This is because the switch user operation is always chained to the subsequent operation.

You can use this element to determine who invoked the application.

authority_bitmap - User authorization level monitor element

The authorities granted to the user and to the groups to which the user belongs. These include authorities granted to roles that are granted to the user and to the groups to which the user belongs. Authorities granted to a user or to roles granted to the user are considered user authorities. Authorities granted to a group to which the user belongs or to roles granted to the group to which the user belongs are considered group authorities.

Table 165. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic
Application	appl_info	Basic

Table 166. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

Usage

The authority_bitmap monitor element has the format of an array. Each array element is a single character that represents whether or not the user ID has been granted a specific authority and how the user has received that authority.

Individual array elements are indexed through an index value defined in the `sql.h` file. The value of an index in the authority_bitmap array is called an *authority index*. For example, SQL_DBAUTH_SYSADM is the index to determine if the user has SYSADM authority.

The value of one element in the authority_bitmap array identified by an authority index represents whether the authority is held by an authorization ID. To determine how the authorization ID is held, for each array element identified by the authority index, use the following defines from sql.h:

SQL_AUTH_ORIGIN_USER
If this bit is on, then the authorization ID has the authority granted to the user or to a role granted to the user.

SQL_AUTH_ORIGIN_GROUP
If this bit is on, then the authorization ID has the authority granted to the group or to a role granted to the group.

For example, to determine if a user holds DBADM authority, verify the following value:

```
authority_bitmap[SQL_DBAUTH_DBADM]
```

To determine if the DBADM authority is held directly by the user, verify the following:

```
authority_bitmap[SQL_DBAUTH_DBADM] & SQL_AUTH_ORIGIN_USER
```

authority_lvl - User authorization level monitor element

The highest authority level granted to an application.

Note: The authority_lvl monitor element is deprecated starting with DB2 database Version 9.5. Use the authority_bitmap monitor element instead. See "authority_bitmap - User authorization level monitor element" on page 537.

Table 167. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic
Application	appl_info	Basic

Table 168. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

Usage The operations allowed by an application are granted either directly or indirectly.

The following defines from sql.h may be used to determine the authorizations granted explicitly to a user:

- SQL_SYSADM

- SQL_DBADM
- SQL_CREATETAB
- SQL_BINDADD
- SQL_CONNECT
- SQL_CREATE_EXT_RT
- SQL_CREATE_NOT_FENC
- SQL_SYSCTRL
- SQL_SYSMAINT

The following defines from sql.h may be used to determine indirect authorizations inherited from group or public:

- SQL_SYSADM_GRP
- SQL_DBADM_GRP
- SQL_CREATETAB_GRP
- SQL_BINDADD_GRP
- SQL_CONNECT_GRP
- SQL_CREATE_EXT_RT_GRP
- SQL_CREATE_NOT_FENC_GRP
- SQL_SYSCTRL_GRP
- SQL_SYSMAINT_GRP

auto_storage_hybrid - Hybrid automatic storage table space indicator monitor element

If the table space is an automatic storage table space with some non-automatic storage containers, this monitor element returns a value of 1. Otherwise, it returns a value of 0.

Table 169. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Usage

A hybrid automatic storage table space is a table space that has been converted to be managed by automatic storage using the ALTER TABLESPACE command, but has not yet been rebalanced. This table space still has non-automatic storage containers. After the table space is rebalanced, it contains only automatic storage containers, and is no longer considered a hybrid table space.

automatic - Buffer pool automatic monitor element

Indicates whether a particular buffer pool has self-tuning enabled. This element is set to 1 if self-tuning is enabled for the buffer pool; and 0 otherwise.

Table 170. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE

bin_id - Histogram bin identifier monitor element

The identifier of a histogram bin. The **bin_id** is unique within a histogram.

Table 171. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_histogrambin	-

Usage

Use this element to distinguish bins within the same histogram.

binds_precompiles - Binds/Precompiles Attempted

The number of binds and pre-compiles attempted.

Element identifier
binds_precompiles

Element type
counter

Table 172. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 173. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage You can use this element to gain insight into the current level of activity within the database manager.

This value does not include the count of *int_auto_rebinds*, but it does include binds that occur as a result of the REBIND PACKAGE command.

block_ios - Number of block I/O requests monitor element

The number of block I/O requests. More specifically, the number of times DB2 performs sequential prefetching of pages into the block area of the buffer pool.

Table 174. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 175. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool	Buffer Pool

Usage

If block-based buffer pool is enabled, this monitor element will report how often block I/O is being done. Otherwise, this monitor element will return 0. The number of block I/O requests is monitored only during sequential prefetching when using block-based buffer pools.

If block-based buffer pool is enabled and this number is very low, or close to the number of vectored I/Os (the value of the **vectored_ios** monitor element), consider changing the block size. This state can be an indication of the following:

- The extent size of one or more table spaces bound to the buffer pool is smaller than the block size specified for the buffer pool.
- Some pages requested in the prefetch request are already present in the page area of the buffer pool.

The prefetcher allows some wasted pages in each buffer pool block, but if too many pages are wasted, then the prefetcher will decide to perform vectored I/O into the page area of the buffer pool.

To take full advantage of the sequential prefetch performance improvements that block-based buffer pools provide, it is essential to choose an appropriate value for the block size. This can, however, be difficult because multiple table spaces with different extent sizes can be bound to the same block-based buffer pool. For optimal performance, it is recommended that you bind table spaces with the same extent size to a block-based buffer pool with a block size equal to the extent size. Good performance can be achieved when the extent size of the table spaces are greater than the block size, but not when the extent size is smaller than the block size.

For example, if the extent size is 2 and the block size is 8, vectored I/O would be used instead of block I/O (block I/O would have wasted 6 pages). A reduction of the block size to 2 would solve this problem.

blocking_cursor - Blocking Cursor

This element indicates if the statement being executed is using a blocking cursor.

Element identifier
blocking_cursor

Element type
information

Table 176. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement
DCS Statement	dcs_stmt	Statement

Table 177. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks with Details	event_detailed_dlconn	-
Statements	event_stmt	-

Usage Using blocking for data transfer for a query can improve its performance. The SQL used for a query can affect the use of blocking and might require some modification.

blocks_pending_cleanup - Pending cleanup rolled-out blocks monitor element

The total number of MDC table blocks in the database that are pending asynchronous cleanup following a roll out delete.

Table 178. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	-
Database	event_db	-

Usage

Use this element to determine the number of MDC table blocks that, following the deletion of a defer cleanup roll out, have not been released back to the system as available storage.

bottom - Histogram bin bottom monitor element

The exclusive bottom end of the range of a histogram bin. The value of this monitor element is also the top inclusive end of the range of the previous histogram bin, if there is one.

Table 179. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_histogrambin	-

Usage

Use this element with the corresponding **top** element to determine the range of a bin within a histogram.

boundary_leaf_node_splits - Boundary leaf node splits monitor element

Number of times a boundary leaf node was split during an insert operation.

Table 180. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

bp_cur_buffsz - Current Size of Buffer Pool

The current buffer pool size.

Table 181. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool_nodeinfo	Buffer Pool

bp_id - Buffer pool identifier monitor element

This element contains the buffer pool identifier for the buffer pool that is being monitored.

Table 182. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool	Basic

bp_name - Buffer pool name monitor element

The name of the buffer pool.

Table 183. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE

Table 184. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool	Basic

Usage Each database requires at least one buffer pool. Depending on your needs, you may choose to create several buffer pools, each of a different size, for a single database. The CREATE, ALTER, and DROP BUFFERPOOL statements allow you to create, change, or remove a buffer pool.

When a database is created, it has a default buffer pool called IBMDEFAULTBP with a size determined by the platform. It also has a set of system buffer pools, each corresponding to a different page size:

- IBMSYSTEMBP4K
- IBMSYSTEMBP8K
- IBMSYSTEMBP16K
- IBMSYSTEMBP32K

These system buffer pools cannot be altered.

bp_new_buffsz - New Buffer Pool Size

The size the buffer pool will be changed to once the database is restarted. When the ALTER BUFFERPOOL statement is executed as DEFERRED, the buffer pool size is not changed until the database is stopped and restarted.

Table 185. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool_nodeinfo	Buffer Pool

bp_pages_left_to_remove - Number of Pages Left to Remove

The number of pages left to remove from the buffer pool before the buffer pool resize is completed. This applies only to buffer pool resize operations invoked by ALTER BUFFERPOOL statements executed as IMMEDIATE.

Table 186. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool_nodeinfo	Buffer Pool

bp_tbsp_use_count - Number of Table Spaces Mapped to Buffer Pool

The number of table spaces using this buffer pool.

Table 187. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool_nodeinfo	Buffer Pool

buff_auto_tuning - FCM buffer auto-tuning indicator monitor element

Indicates whether the number of fast communication manager (FCM) buffers is set and tuned automatically. A value of 1 means "Yes"; a value of 0 means "No".

Table 188. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_FCM - Get FCM metrics	Always collected

Usage

FCM buffer auto-tuning is enabled by setting the **fcm_num_buffers** configuration parameter to AUTOMATIC.

buff_free - FCM buffers currently free monitor element

This element indicates the number of FCM buffers currently free.

Table 189. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_FCM - Get FCM metrics	Always collected

Table 190. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	fcm	Basic

Usage

To calculate the percentage of free FCM buffers, use the following formula:

```
(buff_free/buff_total) * 100
```

If the percentage of free FCM buffers falls below 20% and if the FCM buffer auto-tuning is enabled, then DB2 database manager will adjust the FCM buffer numbers.

If the percentage of free FCM buffers falls below 20% and if the FCM buffer auto-tuning is not enabled, then you need to tune the **fcm_num_buffers** configuration parameter.

buff_free_bottom - Minimum FCM buffers free monitor element

The lowest number of free FCM buffers reached during processing.

Table 191. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_FCM - Get FCM metrics	Always collected

Table 192. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	fcm	Basic

Usage

Use this element along with the **fcm_num_buffers** configuration parameter to determine the maximum FCM buffer pool utilization. If the value of the **buff_free_bottom** monitor element is low, increase the value of the **fcm_num_buffers** configuration parameter to ensure that operations do not run out of FCM buffers. If the value of the **buff_free_bottom** monitor element is high, decrease the value of the **fcm_num_buffers** configuration parameter to conserve system resources.

buff_max - Maximum possible number of FCM buffers monitor element

Maximum number of fast communication manager (FCM) buffers that can be allocated, based on the amount of virtual memory reserved when the instance was started.

Table 193. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_FCM - Get FCM metrics	Always collected

Table 194. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	fcm	Basic

Usage

This internal monitor element is used by IBM Support only.

buff_total - Number of currently allocated FCM buffers monitor element

Number of fast communication manager (FCM) buffers currently allocated. This number includes both in-use buffers and free buffers.

Table 195. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_FCM - Get FCM metrics	Always collected

Table 196. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	fcm	Basic

Usage

If the **buff_auto_tuning** monitor element indicates that FCM is being tuned automatically, then the value of the **buff_total** monitor element is adjusted based on the demand for FCM buffers.

To determine the number of FCM buffers currently in use, use the following formula:

```
buff_total - buff_free
```

To calculate the percentage of free FCM buffers, use the following formula:

```
(buff_free/buff_total) * 100
```

If the percentage of free FCM buffers falls below 20% and if the FCM buffer auto-tuning is enabled, then DB2 database manager will adjust the FCM buffer numbers.

If the percentage of free FCM buffers falls below 20% and if the FCM buffer auto-tuning is not enabled, then you need to tune the **fcm_num_buffers** configuration parameter.

byte_order - Byte Order of Event Data

The byte ordering of numeric data, specifically whether the event data stream was generated on a "big endian" server (for example, a RS/6000®) or "little endian" server (for example, an Intel-based PC running Windows 2000).

Table 197. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Event Log Header	event_log_header	-

Usage This information is needed to allow you to interpret numeric data in the data stream, since the byte order of integers on a "big endian" server is the reverse of the byte order on a "little endian" server.

If the application that processes the data recognizes that it is running on one type of computer hardware (for example, a big endian computer), while the event data was produced on the other type of computer hardware (for example, a little endian computer), then the monitoring application will have to reverse the bytes of numeric data fields before interpreting them. Otherwise, byte reordering is not required.

This element can be set to one of the following API constants:

- SQLM_BIG_ENDIAN
- SQLM_LITTLE_ENDIAN

cat_cache_inserts - Catalog cache inserts monitor element

The number of times that the system tried to insert table descriptor or authorization information into the catalog cache.

Table 198. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 199. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 200. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

In conjunction with "Catalog Cache Lookups", you can calculate the catalog cache hit ratio using the following formula:

```
1 - (Catalog Cache Inserts / Catalog Cache Lookups)
```

See the **cat_cache_lookups** monitor element for more information on using this element.

cat_cache_lookups - Catalog cache lookups monitor element

The number of times that the catalog cache was referenced to obtain table descriptor information or authorization information.

Table 201. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 201. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 202. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 203. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

This element includes both successful and unsuccessful accesses to the catalog cache. The catalog cache is referenced whenever:

- a table, view, or alias name is processed during the compilation of an SQL statement
- database authorization information is accessed
- a routine is processed during the compilation of an SQL statement

To calculate the catalog cache hit ratio use the following formula:

```
(1 - (cat_cache_inserts / cat_cache_lookups))
```

indicates how well the catalog cache is avoiding catalog accesses. If the ratio is high (more than 0.8), then the cache is performing well. A smaller ratio might suggest that the **catalogcache_sz** configuration parameter should be increased. You should expect a large ratio immediately following the first connection to the database.

The execution of Data Definition Language (DDL) SQL statements involving a table, view, or alias will evict the table descriptor information for that object from the catalog cache causing it to be re-inserted on the next reference. In addition, GRANT and REVOKE statements for database authorization and execute privilege of routines will evict the subject authorization information from the catalog cache. Therefore, the heavy use of DDL statements and GRANT/REVOKE statements may also increase the ratio.

cat_cache_overflows - Catalog Cache Overflows

The number of times that the catalog cache overflowed the bounds of its allocated memory.

Table 204. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 205. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage

Use this element with the**cat_cache_size_top** monitor element to determine whether the size of the catalog cache needs to be increased to avoid overflowing.

Catalog cache space is reclaimed by evicting table descriptor information for tables, views, or aliases, or authorization information that is not currently in use by any transaction.

If the value of the **cat_cache_overflows** monitor element is large, the catalog cache may be too small for the workload. Enlarging the catalog cache may improve its performance. If the workload includes transactions which compile a large number of SQL statements referencing many tables, views, aliases, user-defined functions, or stored procedures in a single unit of work, then compiling fewer SQL statements in a single transaction may improve the performance of the catalog cache. Or if the workload includes binding of packages containing many SQL statements referencing many tables, views, aliases, user-defined functions, or stored

procedures, you can try splitting packages so that they include fewer SQL statements to improve performance.

cat_cache_size_top - Catalog cache high watermark monitor element

The largest logical size reached by the catalog cache.

Table 206. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Table 207. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage

This element indicates the maximum number of bytes the catalog cache required logically for the workload run against the database since it was activated.

The catalog cache is managed by logical size, which does not include memory management overhead. The **pool_watermark** element in the database snapshot provides the physical high water mark value for memory used by the catalog cache. The logical size rather than physical size should be used for catalog cache monitoring and tuning efforts.

If the catalog cache overflowed, then this element contains the largest size reached by the catalog cache during the overflow. Check the **cat_cache_overflows** monitor element to determine if such a condition occurred.

You can determine the minimum size of the catalog cache required by your workload by:

```
maximum catalog cache size / 4096
```

Rounding the result up to a whole number, indicates the minimum number of 4K pages required by the catalog cache to avoid overflow.

catalog_node - Catalog Node Number

The node number of the node where the database catalog tables are stored.

Element identifier
catalog_node

Element type
information

Table 208. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Table 209. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage The catalog node is the node where all system catalog tables are stored. All access to system catalog tables must go through this node.

catalog_node_name - Catalog Node Network Name

The network name of the catalog node.

Element identifier
catalog_node_name

Element type
information

Table 210. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Table 211. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage Use this element to determine the location of a database.

ch_auto_tuning - FCM channel auto-tuning indicator monitor element

Indicates whether the number of fast communication manager (FCM) channels is set and tuned automatically. A value of 1 means "Yes"; a value of 0 means "No".

Table 212. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_FCM - Get FCM metrics	Always collected

Usage

FCM channel auto-tuning is enabled by setting the **fcm_num_channels** configuration parameter to AUTOMATIC.

ch_free - Channels currently free monitor element

This element indicates the number of FCM communication channels that are currently free.

Table 213. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_FCM - Get FCM metrics	Always collected

Table 214. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	fcm	Basic

Usage

To calculate the percentage of free FCM channels, use the following formula:
`(ch_free/ch_total) * 100`

If the percentage of free FCM channels falls below 20% and if the FCM channel auto-tuning is enabled, then DB2 database manager will adjust the FCM channel numbers.

If the percentage of free FCM channels falls below 20% and if the FCM channel auto-tuning is not enabled, then you need to tune the **fcm_num_channels** configuration parameter.

ch_free_bottom - Minimum channels free monitor element

The lowest number of free FCM communication channels reached during processing.

Table 215. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_FCM - Get FCM metrics	Always collected

Table 216. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	fcm	Basic

Usage

Use this monitor element along with the **fcm_num_channels** configuration parameter to determine the maximum connection entry utilization.

ch_max - Maximum possible number of FCM channels monitor element

Maximum number of fast communication manager (FCM) channels that can be allocated based on the amount of virtual memory reserved when the instance was started.

Table 217. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_FCM - Get FCM metrics	Always collected

Table 218. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	fcm	Basic

Usage

This internal monitor element is used by IBM Support only.

ch_total - Number of currently allocated FCM channels monitor element

Number of fast communication manager (FCM) channels currently allocated. This number includes both in-use channels and free channels.

Table 219. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_FCM - Get FCM metrics	Always collected

Table 220. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	fcm	Basic

Usage

If the **ch_auto_tuning** monitor element indicates that FCM is being tuned automatically, then the value of the **ch_total** monitor element is adjusted based on the demand for FCM channels.

To determine the number of FCM channels currently in use, use the following formula:

```
ch_total - ch_free
```

To calculate the percentage of free FCM channels, use the following formula:

```
(ch_free/ch_total) * 100
```

If the percentage of free FCM channels falls below 20% and if the FCM channel auto-tuning is enabled, then DB2 database manager will adjust the FCM channel numbers.

If the percentage of free FCM channels falls below 20% and if the FCM channel auto-tuning is not enabled, then you need to tune the **fcm_num_channels** configuration parameter.

client_acctng - Client accounting string monitor element

The data passed to the target database for logging and diagnostic purposes, if the sqleseti API was issued in this connection. The current value of the CLIENT_ACCTNG special register for this connection, unit of work, or activity.

Note: This element is reported for the coordinating member only. On remote members, the value reported is a string with a length of 0.

This monitor element is synonymous to the **tpmon_acc_str** monitor element. The **client_acctng** monitor element is used for monitoring table functions and event monitors that write to unformatted tables, which were introduced in DB2 Version 9.7. The **tpmon_acc_str** monitor element is used for snapshot monitors and event monitors that write to tables, files, and pipes.

Table 221. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	Always collected
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 222. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-
Threshold violations	event_thresholdviolations	-

Usage

Use this element for problem determination and accounting purposes.

client_applname - Client application name monitor element

Identifies the server transaction program performing the transaction, if the sqleseti API was issued in this connection. The current value of the CLIENT_APPLNAME special register for this connection, unit of work, or activity.

Note: This element is reported for the coordinating member only. On remote members, the value reported is a string with a length of 0.

This monitor element is synonymous to the **tpmon_client_app** monitor element. The **client_applname** monitor element is used for monitoring table functions and event monitors that write to unformatted tables, which were introduced in DB2 Version 9.7. The **tpmon_client_app** monitor element is used for snapshot monitors and event monitors that write to tables, files, and pipes.

Table 223. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected

Table 223. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected
MON_GET_CONNECTION table function - Get connection metrics	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected

Table 224. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-
Threshold violations	event_thresholdviolations	-

Usage

Use this element for problem determination and accounting purposes.

client_db_alias - Database Alias Used by Application

The alias of the database provided by the application to connect to the database.

Element identifier
client_db_alias

Element type
information

Table 225. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_id_info	Basic
Lock	appl_lock_list	Basic

Table 226. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connections	event_connheader	-

Usage This element can be used to identify the actual database that the application is accessing. The mapping between this name and *db_name* could be done by using the database directories at the client node and the database manager server node.

This is the alias defined within the database manager where the database connection request originated.

This element can also be used to help you determine the authentication type, since different database aliases can have different authentication types.

client_hostname - Client hostname monitor element

The hostname of the machine the client application is connecting from.

Table 227. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected

Table 228. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-
Threshold violations	event_thresholdviolations	-

client_idle_wait_time - Client idle wait time monitor element

This monitor element records time spent waiting for the client to send its next request. The value is given in milliseconds.

Table 229. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE

Table 230. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this monitor element to determine the amount of time spent waiting for requests from a client, as opposed to working on requests. A high client idle time may indicate performance issues that need to be addressed on the client rather than the server.

client_pid - Client process ID monitor element

The process ID of the client application that made the connection to the database.

Table 231. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	Always collected
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected

Table 232. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
Application	appl	Basic
DCS Application	dcs_appl_info	Basic

Table 233. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Threshold violations	event_thresholdviolations	-
Unit of work	-	-
Connections	event_connheader	-

Usage

You can use this element to correlate monitor information such as CPU and I/O time to your client application.

In the case of a DRDA AS connection, this element will be set to 0.

client_platform - Client operating platform monitor element

The operating system on which the client application is running.

Table 234. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	Always collected
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected

Table 235. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
Application	appl	Basic
DCS Application	dcs_appl_info	Basic

Table 236. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-
Connections	event_connheader	-
Threshold violations	event_thresholdviolations	-

Usage

This element can be used for problem determination for remote applications. Values for this field is one of:

SQLM_PROT_UNKNOWN
The client is communicating using an unknown protocol. This value will only be returned if future clients connect with an earlier level of the server.

SQLM_PROT_LOCAL
The client is running on the same node as the server and no communications protocol is in use.

SQLM_PROT_TCPIP
TCP/IP

SQLM_PROT_TCPIP4
TCP/IP4

SQLM_PROT_TCPIP6
TCP/IP6

SQLM_PROT_IPXSPX
IPX/SPX

SQLM_PROT_NPIPE
Named Pipe Protocol

client_port_number - Client port number monitor element

For TCP/IP connections, the port number on the client machine the application is using to communicate with the database server.

Table 237. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected

Table 238. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-
Threshold violations	event_thresholdviolations	-

client_prdid - Client product and version ID monitor element

The product and version that is running on the client.

Table 239. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	Always collected
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected

Table 240. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_id_info	Basic
DCS Application	dcs_appl_info	Basic

Table 241. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-
Connections	event_connheader	-
Threshold violations	event_thresholdviolations	-

Usage

You can use this element to identify the product and code version of the IBM data server client. It is in the form PPPVVRRM, where:

- PPP identifies the product, which is "SQL" for the DB2 products
- VV identifies a 2-digit version number (with high-order 0 in the case of a 1-digit version)
- RR identifies a 2-digit release number (with high-order 0 in the case of a 1-digit release)
- M identifies a 1-character modification level (0-9 or A-Z).

client_protocol - Client communication protocol monitor element

The communication protocol that the client application is using to communicate with the server.

Table 242. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	Always collected
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected

Table 243. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
Application	appl	Basic
DCS Application	dcs_appl_info	Basic

Table 244. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-
Connections	event_connheader	-
Threshold violations	event_thresholdviolations	-

Usage

This element can be used for problem determination for remote applications. Values for this field are:

SQLM_PROT_UNKNOWN
The client is communicating using an unknown protocol. This value will only be returned if future clients connect with an earlier level of the server.

SQLM_PROT_LOCAL
The client is running on the same node as the server and no communications protocol is in use.

SQLM_PROT_TCPIP
TCP/IP

SQLM_PROT_TCPIP4
TCP/IP4

SQLM_PROT_TCPIP6
TCP/IP6

SQLM_PROT_IPXSPX
IPX/SPX

SQLM_PROT_NPIPE
Named Pipe Protocol

If using the APPLICATIONS administrative view this monitor element returns a text identifier, value for this field is one of:

- LOCAL
- NPIPE
- TCPIP
- TCPIP4
- TCPIP6

client_userid - Client user ID monitor element

The client user ID generated by a transaction manager and provided to the server, if the sqleseti API is used. The current value of the CLIENT_USERID special register for this connection, unit of work, or activity.

Note: This element is reported for the coordinating member only. On remote members, the value reported is a string with a length of 0.

This monitor element is synonymous to the **tpmon_client_userid** monitor element. The **client_userid** monitor element is used for monitoring table functions and event monitors that write to unformatted tables, which were introduced in DB2 Version 9.7. The **tpmon_client_userid** monitor element is used for snapshot monitors and event monitors that write to tables, files, and pipes.

Table 245. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	Always collected
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 246. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-
Threshold violations	event_thresholdviolations	-

Usage

Use this element in application server or Transaction Processing monitor environments to identify the end-user for whom the transaction is being executed.

client_wrkstnname - Client workstation name monitor element

Identifies the client's system or workstation (for example CICS EITERMID), if the sqleseti API was issued in this connection. The current value of the CLIENT_WRKSTNNAME special register for this connection, unit of work, or activity.

Note: This element is reported for the coordinating member only. On remote members, the value reported is a string with a length of 0.

This monitor element is synonymous to the **tpmon_client_wkstn** monitor element. The **client_wrkstnname** monitor element is used for monitoring table functions and event monitors that write to unformatted tables, which were introduced in DB2 Version 9.7. The **tpmon_client_wkstn** monitor element is used for snapshot monitors and event monitors that write to tables, files, and pipes.

Table 247. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	Always collected
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 248. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-
Threshold violations	event_thresholdviolations	-

Usage

Use this element to identify the user's machine by node ID, terminal ID, or similar identifiers.

codepage_id - ID of Code Page Used by Application

The code page identifier.

Table 249. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_id_info	Basic
Lock	appl_lock_list	Basic
DCS Application	dcs_appl_info	Basic

Table 250. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Event Log Header	event_log_header	-
Connections	event_connheader	-

Usage For snapshot monitor data, this is the code page at the partition where the monitored application started. This identifier may be used for problem determination for remote applications. You may use this information to ensure that data conversion is supported between the application code page and the database code page (or for DRDA host databases, the host CCSID). For information about supported code pages, see the *Administration Guide*.

For event monitor data, this is the code page of the database for which event data is collected. You can use this element to determine whether your event monitor application is running under a different code page from that used by the database. Data written by the event monitor uses the database code page. If your event monitor application uses a different code page, you may need to perform some character conversion to make the data readable.

comm_private_mem - Committed Private Memory

The amount of private memory that the instance of the database manager has currently committed at the time of the snapshot. The comm_private_mem value returned is only relevant on Windows operating systems.

Table 251. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

commit_sql_stmts - Commit Statements Attempted

The total number of SQL COMMIT statements that have been attempted.

Table 252. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Database	dbase_remote	Basic
Application	appl	Basic
Application	appl_remote	Basic
DCS Database	dcs_dbase	Basic
DCS Application	dcs_appl	Basic

For snapshot monitoring, this counter can be reset.

Table 253. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage A small rate of change in this counter during the monitor period may indicate that applications are not doing frequent commits, which may lead to problems with logging and data concurrency.

You can also use this element to calculate the total number of units of work by calculating the sum of the following:

```
  commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks
```

Note: The units of work calculated will only include those since the later of:

- The connection to the database (for database-level information, this is the time of the first connection)
- The last reset of the database monitor counters.

This calculation can be done at a database or application level.

comp_env_desc - Compilation environment monitor element

This element stores information about the compilation environment used when compiling the SQL statement.

Table 254. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected

Table 254. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	Always collected

Table 255. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks with Details History Values	event_stmt_history	-
Deadlocks with Details History	event_stmt_history	-
Activities	event_activitystmt	-
Package cache	-	COLLECT BASE DATA

Table 256. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Package cache	-	COLLECT BASE DATA

Usage

This monitor element stores the compilation environment description in a binary large object. To view this information in readable form, use the COMPILATION_ENV table function.

You can provide this element as input to the COMPILATION_ENV table function, or to the SET COMPILATION ENVIRONMENT SQL statement.

completion_status - Completion status monitor element

The status of the unit of work.

Table 257. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-

Usage

Use this element to determine if the unit of work ended due to a deadlock or abnormal termination. The possible values are listed in the `sqllib/misc/DB2EvmonUOW.xsd` file:

- UNKNOWN
- COMMIT
- ROLLBACK
- GLOBAL_COMMIT
- GLOBAL_ROLLBACK
- XA_END
- XA_PREPARE

con_elapsed_time - Most Recent Connection Elapsed Time

The elapsed time that the DCS application that most recently disconnected from this host database was connected.

Element identifier
con_elapsed_time

Element type
time

Table 258. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Timestamp

Usage

Use this element as an indicator of the length of time that applications are maintaining connections to a host database.

This element is composed of two subelements that report time spent as seconds and microseconds (one millionth of a second). The names of the subelements can be derived by adding "_s" and "_ms" to the name of this monitor element. To retrieve the total time spent for this monitor element, the values of the two subelements must be added together. For example, if the "_s" subelement value is 3 and the "_ms" subelement value is 20, then the total time spent for the monitor element is 3.00002 seconds.

con_local_dbases - Local Databases with Current Connects

The number of local databases that have applications connected.

Table 259. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

Usage This value gives an indication of how many database information records you can expect when gathering data at the database level.

The applications can be running locally or remotely, and may or may not be executing a unit of work within the database manager

con_response_time - Most Recent Response Time for Connect

The elapsed time between the start of connection processing and actual establishment of a connection, for the most recent DCS application that connected to this database.

Element identifier
con_response_time

Element type
time

Table 260. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Timestamp

Usage

Use this element as an indicator of the time it currently takes applications to connect to a particular host database.

This element is composed of two subelements that report time spent as seconds and microseconds (one millionth of a second). The names of the subelements can be derived by adding "_s" and "_ms" to the name of this monitor element. To retrieve the total time spent for this monitor element, the values of the two subelements must be added together. For example, if the "_s" subelement value is 3 and the "_ms" subelement value is 20, then the total time spent for the monitor element is 3.00002 seconds.

concurrent_act_top - Concurrent activity top monitor element

The high watermark for the concurrent activities (at any nesting level) in a service subclass since the last reset.

Note: This element monitors the highest concurrent execution of all activities, including those activities that do not participate in the CONCURRENTDBCOORDACTIVITIES threshold. For example, although CALL statements do not count towards the concurrency that is enforced by the CONCURRENTDBCOORDACTIVITIES threshold, they are included in the concurrent activity high watermark measurement.

Table 261. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-

Usage

Use this element to know the highest concurrency of activities (including nested activities) reached on a partition for a service subclass in the time interval collected.

concurrent_connection_top - Concurrent connection top monitor element

High watermark for concurrent coordinator connections in this service class since the last reset. This field has the same value in every subclass of the same superclass.

Table 262. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-

Usage

This element may be useful in determining where to place thresholds on connection concurrency by showing where the current high watermark is. It is also useful for verifying that such a threshold is configured correctly and doing its job.

concurrent_wlo_act_top - Concurrent WLO activity top monitor element

High watermark for concurrent activities (at any nesting level) of any occurrence of this workload since the last reset.

Table 263. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wlstats	-

Usage

Use this element to know the highest number of concurrent activities reached on a partition for any occurrence of this workload in the time interval collected.

concurrent_wlo_top - Concurrent workload occurrences top monitor element

The high watermark for the concurrent occurrences of a workload since the last reset.

Table 264. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wlstats	-
Statistics	event_scstats	-

Usage

Use this element to know the highest concurrency of workload occurrences reached on a partition for a workload in the time interval collected.

concurrentdbcoordactivities_db_threshold_id - Concurrent database coordinator activities database threshold ID monitor element

The ID of the CONCURRENTDBCOORDACTIVITIES database threshold that was applied to the activity.

Table 265. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand which CONCURRENTDBCOORDACTIVITIES database threshold, if any, was applied to the activity.

concurrentdbcoordactivities_db_threshold _queued - Concurrent database coordinator activities database threshold queued monitor element

This monitor element returns 'Yes' to indicate that the activity was queued by the CONCURRENTDBCOORDACTIVITIES database threshold. 'No' indicates that the activity was not queued.

Table 266. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand if the activity was queued by the CONCURRENTDBCOORDACTIVITIES database threshold applied to the activity.

concurrentdbcoordactivities_db_threshold _value - Concurrent database coordinator activities database threshold value monitor element

This monitor element returns the upper bound of the CONCURRENTDBCOORDACTIVITIES database threshold that was applied to the activity.

Table 267. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand the value of the CONCURRENTDBCOORDACTIVITIES database threshold applied to the activity, if any.

concurrentdbcoordactivities_db_threshold _violated - Concurrent database coordinator activities database threshold violated monitor element

This monitor element returns 'Yes' to indicate that the activity violated the CONCURRENTDBCOORDACTIVITIES database threshold. 'No' indicates that the activity has not yet violated the threshold.

Table 268. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to determine if the activity violated the CONCURRENTDBCOORDACTIVITIES database threshold that was applied to the activity.

concurrentdbcoordactivities_subclass_threshold _id - Concurrent database coordinator activities service subclass threshold ID monitor element

This monitor element returns the ID of the CONCURRENTDBCOORDACTIVITIES service subclass threshold threshold that was applied to the activity.

Table 269. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand which CONCURRENTDBCOORDACTIVITIES service subclass threshold, if any, was applied to the activity.

concurrentdbcoordactivities_subclass_ threshold_queued - Concurrent database coordinator activities service subclass threshold queued monitor element

This monitor element returns 'Yes' to indicate that the activity was queued by the CONCURRENTDBCOORDACTIVITIES service subclass threshold. 'No' indicates that the activity was not queued.

Table 270. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand if the activity was queued by the CONCURRENTDBCOORDACTIVITIES service subclass threshold applied to the activity.

concurrentdbcoordactivities_subclass_ threshold_value - Concurrent database coordinator activities service subclass threshold value monitor element

This monitor element returns the upper bound of the CONCURRENTDBCOORDACTIVITIES service subclass threshold that was applied to the activity.

Table 271. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand the value of the CONCURRENTDBCOORDACTIVITIES service subclass threshold applied to the activity, if any.

concurrentdbcoordactivities_subclass_ threshold_violated - Concurrent database coordinator activities service subclass threshold violated monitor element

This monitor element returns 'Yes' to indicate that the activity violated the CONCURRENTDBCOORDACTIVITIES service subclass threshold. 'No' indicates that the activity has not yet violated the threshold.

Table 272. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to determine if the activity violated the CONCURRENTDBCOORDACTIVITIES service subclass threshold that was applied to the activity.

concurrentdbcoordactivities_superclass_ threshold_id - Concurrent database coordinator activities service superclass threshold ID monitor element

The ID of the CONCURRENTDBCOORDACTIVITIES_SUPERCLASS threshold that was applied to the activity.

Table 273. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand which CONCURRENTDBCOORDACTIVITIES service superclass threshold, if any, was applied to the activity.

concurrentdbcoordactivities_superclass_ threshold_queued - Concurrent database coordinator activities service superclass threshold queued monitor element

This monitor element returns 'Yes' to indicate that the activity was queued by the CONCURRENTDBCOORDACTIVITIES service superclass threshold. 'No' indicates that the activity was not queued.

Table 274. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand if the activity was queued by the CONCURRENTDBCOORDACTIVITIES service superclass threshold applied to the activity.

concurrentdbcoordactivities_superclass_ threshold_value - Concurrent database coordinator activities service superclass threshold value monitor element

The upper bound of the CONCURRENTDBCOORDACTIVITIES service superclass threshold that was applied to the activity.

Table 275. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand the value of the CONCURRENTDBCOORDACTIVITIES service superclass threshold applied to the activity, if any.

concurrentdbcoordactivities_superclass_ threshold_violated - Concurrent database coordinator activities service superclass threshold violated monitor element

This monitor element returns 'Yes' to indicate that the activity violated the CONCURRENTDBCOORDACTIVITIES service superclass threshold. 'No' indicates that the activity has not yet violated the threshold.

Table 276. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to determine if the activity violated the CONCURRENTDBCOORDACTIVITIES service superclass threshold that was applied to the activity.

concurrentdbcoordactivities_wl_was_threshold _id - Concurrent database coordinator activities workload work action set threshold ID monitor element

The identifier of the CONCURRENTDBCOORDACTIVITIES workload work action set threshold that was applied to the activity.

Table 277. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand which CONCURRENTDBCOORDACTIVITIES workload work action set threshold, if any, was applied to the activity.

concurrentdbcoordactivities_wl_was_threshold _queued - Concurrent database coordinator activities workload work action set threshold queued monitor element

This monitor element returns 'Yes' to indicate that the activity was queued by the CONCURRENTDBCOORDACTIVITIES workload work action set threshold. 'No' indicates that the activity was not queued.

Table 278. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand if the activity was queued by the CONCURRENTDBCOORDACTIVITIES workload work action set threshold applied to the activity.

concurrentdbcoordactivities_wl_was_threshold _value - Concurrent database coordinator activities workload work action set threshold value monitor element

The upper bound of the CONCURRENTDBCOORDACTIVITIES workload work action set threshold that was applied to the activity.

Table 279. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand the value of the CONCURRENTDBCOORDACTIVITIES workload work action set threshold applied to the activity.

concurrentdbcoordactivities_wl_was_threshold _violated - Concurrent database coordinator activities workload work action set threshold violated monitor element

This monitor element returns 'Yes' to indicate that the activity violated the CONCURRENTDBCOORDACTIVITIES workload work action set threshold. 'No' indicates that the activity has not yet violated the threshold.

Table 280. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to determine if the activity violated the CONCURRENTDBCOORDACTIVITIES workload work action set threshold that was applied to the activity.

concurrentdbcoordactivities_work_action_set_ threshold_id - Concurrent database coordinator activities work action set threshold ID monitor element

The ID of the CONCURRENTDBCOORDACTIVITIES work action set threshold that was applied to the activity.

Table 281. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand which CONCURRENTDBCOORDACTIVITIES work action set threshold, if any, was applied to the activity.

concurrentdbcoordactivities_work_action_set_ threshold_queued - Concurrent database coordinator activities work action set threshold queued monitor element

This monitor element returns 'Yes' to indicate that the activity was queued by the CONCURRENTDBCOORDACTIVITIES_WORK_ACTION_SET threshold. 'No' indicates that the activity was not queued.

Table 282. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand if the activity was queued by the CONCURRENTDBCOORDACTIVITIES_WORK_ACTION_SET threshold applied to the activity.

concurrentdbcoordactivities_work_action_set_ threshold_value - Concurrent database coordinator activities work action set threshold value monitor element

The upper bound of the CONCURRENTDBCOORDACTIVITIES_WORK _ACTION_SET threshold that was applied to the activity.

Table 283. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand the value of the CONCURRENTDBCOORDACTIVITIES_WORK threshold applied to the activity, if any.

concurrentdbcoordactivities_work_action_set_ threshold_violated - Concurrent database coordinator activities work action set threshold violated monitor element

This monitor element returns 'Yes' to indicate that the activity violated the CONCURRENTDBCOORDACTIVITIES_WORK_ACTION_SET threshold. 'No' indicates that the activity has not yet violated the threshold.

Table 284. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to determine if the activity violated the CONCURRENTDBCOORDACTIVITIES_WORK_ACTION_SET threshold that was applied to the activity.

conn_complete_time - Connection Request Completion Timestamp

The date and time that a connection request was granted.

Element identifier
conn_complete_time

Element type
timestamp

Table 285. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Timestamp

Usage Use this element to determine when a connection request to the database was granted.

conn_time - Time of database connection monitor element

The date and time of the connection to the database (at the database level, this is the first connection to the database), or when the activate database was issued.

Table 286. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-
Database	event_dbheader	-
Connections	event_connheader	-

Usage

Use this element with the **disconn_time** monitor element to calculate the elapsed time since:

- The database was active (for information at the database level).
- The connection was active (for information at the connection level).

connection_start_time - Connection start time monitor element

The time at which the connection was established with the database server. The connection_time monitor element is an alias of the connection_start_time monitor element.

Table 287. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	Always collected
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected

Table 288. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Threshold violations	event_thresholdviolations	-

connection_status - Connection status monitor element

For snapshot monitor, this monitor element reports the status of the communication connection between the node issuing the GET SNAPSHOT command and other nodes listed in the `db2nodes.cfg` file. For table function monitor, this monitor element reports the text identifier indicating the FCM connection status.

Table 289. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_FCM_CONNECTION_LIST - Get details for all FCM connections	Always collected

Table 290. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	fcm_node	Basic

Usage

For the snapshot monitor, the connection values are :

SQLM_FCM_CONNECT_INACTIVE
No current connection

SQLM_FCM_CONNECT_ACTIVE
Connection is active

For table function monitoring, the following values are available:

Active No current connection

Inactive
Connection is active

Two members can be active, but the communication connection between them remains inactive until there is some communication between the members.

connections_top - Maximum Number of Concurrent Connections

The highest number of simultaneous connections to the database since the database was activated.

Element identifier
connections_top

Element type
watermark

Table 291. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Table 292. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage You may use this element to evaluate the setting of the *maxappls* configuration parameter.

If the value of this element is the same as the *maxappls* parameter, it is likely that some database connection requests were rejected, since *maxappls* limits the number of database connections allowed.

The current number of connections at the time the snapshot was taken can be calculated using the following formula:

```
rem_cons_in + local_cons
```

consistency_token - Package consistency token monitor element

For a given package name and creator, there can exist (starting in DB2 Version 8) multiple versions. The package consistency token helps to identify the version of the package that contains the SQL currently executing.

Table 293. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement

Table 294. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Statements	event_stmt	-

Usage

You can use this element to help identify the package and the SQL statement that is executing.

container_accessible - Accessibility of container monitor element

This element indicates whether a container is accessible. A value of 1 means "Yes"; a value of 0 means "No".

Table 295. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE

Table 296. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_container	Basic

Usage This element can be used in conjunction with the elements **container_id**, **container_name**, **container_type**, **container_total_pages**, **container_usable_pages**, and **container_stripe_set** to describe the container.

container_id - Container identification monitor element

An integer that uniquely defines a container within a table space.

Table 297. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONTAINER table function - Get container metrics	DATA OBJECT METRICS BASE

Table 298. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_container	Basic

Usage This element can be used in conjunction with the elements container_name, container_type, container_total_pages, container_usable_pages, container_stripe_set, and container_accessible to describe the container.

container_name - Container name monitor element

The name of a container.

Table 299. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONTAINER table function - Get container metrics	DATA OBJECT METRICS BASE

Table 300. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_container	Basic

Usage This element can be used in conjunction with the elements container_id, container_type, container_total_pages, container_usable_pages, container_stripe_set, and container_accessible to describe the container.

container_stripe_set - Container stripe set monitor element

The stripe set that a container belongs to.

Table 301. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE

Table 302. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_container	Basic

Usage

Use this monitor element in conjunction with the elements **container_id**, **container_name**, **container_type**, **container_total_pages**, **container_usable_pages**, and **container_accessible** to describe the container. This is only applicable to a DMS table space.

container_total_pages - Total pages in container monitor element

The total number of pages occupied by the container.

Table 303. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE

Table 304. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_container	Basic (DMS table spaces) Buffer Pool (SMS table spaces)

Usage This element can be used in conjunction with the elements container_id, container_name, container_type, container_usable_pages, container_stripe_set, and container_accessible to describe the container.

container_type - Container type monitor element

The type of the container.

Table 305. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONTAINER table function - Get container metrics	DATA OBJECT METRICS BASE

Table 306. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_container	Basic

Usage

This element returns the type of the container, which can be a directory path (for SMS only), file (for DMS) or a raw device (for DMS). This element can be used in conjunction with the elements **container_id**, **container_name**, **container_total_pages**, **container_usable_pages**, **container_stripe_set**, and **container_accessible** to describe the container.

The valid values for this monitor element are defined in the `sqlutil.h` file.

container_usable_pages - Usable pages in container monitor element

The total number of usable pages in a container.

Table 307. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE

Table 308. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_container	Basic (DMS table spaces) Buffer Pool (SMS table spaces)

Usage This element can be used in conjunction with the elements container_id, container_name, container_type, container_total_pages, container_stripe_set, and container_accessible to describe the container. For SMS table spaces, this value is the same as container_total_pages.

coord_act_aborted_total - Coordinator activities aborted total monitor element

The total number of coordinator activities at any nesting level that completed with errors since the last reset. For service classes, the value is updated when the activity completes. For workloads, the value is updated by each workload occurrence at the end of its unit of work.

For service classes, if you remap an activity to a different subclass with a REMAP ACTIVITY action before it aborts, then this activity counts only towards the total of the subclass it aborts in.

Table 309. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wlstats	-

Usage

Use this element to understand if activities on the system are completing successfully. Activities may be aborted due to cancellation, errors or reactive thresholds.

coord_act_completed_total - Coordinator activities completed total monitor element

The total number of coordinator activities at any nesting level that completed successfully since the last reset. For service classes, the value is updated when the activity completes. For workloads, the value is updated by each workload occurrence at the end of its unit of work.

For service classes, if you remap an activity to a different subclass with a REMAP ACTIVITY action before it completes, then this activity counts only towards the total of the subclass it completes in.

Table 310. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wlstats	-
Statistics	event_scstats	-

Usage

This element can be used to determine the throughput of activities in the system or to aid in calculating average activity lifetime across multiple partitions.

coord_act_est_cost_avg - Coordinator activity estimated cost average monitor element

Arithmetic mean of the estimated costs for coordinator DML activities at nesting level 0 associated with this service subclass or work class since the last reset. If the internally tracked average has overflowed, the value -2 is returned. For service subclasses, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service subclass is set to NONE or BASE. For work classes, this monitor elements returns -1 if no COLLECT AGGREGATE ACTIVITY DATA EXTENDED work action is specified for the work class. For workloads, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the workload is set to NONE or BASE. Units are timerons.

For service classes, the estimated cost of an activity is counted only towards the service subclass in which the activity enters the system. When you remap activities

between service subclasses with a REMAP ACTIVITY action, the coord_act_est_cost_avg mean of the service subclass you remap an activity to is unaffected.

Table 311. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wcstats	-
Statistics	event_wlstats	-

Usage

Use this statistic to determine the arithmetic mean of the estimated costs of coordinator DML activities at nesting level 0 that are associated this service subclass, workload, or work class that completed or aborted since the last statistics reset.

This average can also be used to determine whether or not the histogram template used for the activity estimated cost histogram is appropriate. Compute the average activity estimated cost from the activity estimated cost histogram. Compare the computed average with this monitor element. If the computed average deviates from the true average reported by this monitor element, consider altering the histogram template for the activity estimated cost histogram, using a set of bin values that are more appropriate for your data.

coord_act_exec_time_avg - Coordinator activities execution time average monitor element

Arithmetic mean of execution times for coordinator activities at nesting level 0 associated with this service subclass or work class since the last reset. If the internally tracked average has overflowed, the value -2 is returned. For service subclasses, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service subclass is set to NONE. For work classes, this monitor elements returns -1 if no COLLECT AGGREGATE ACTIVITY DATA work action is specified for the work class. For workloads, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the workload is set to NONE. Units are milliseconds.

For service classes, when you remap activities between service subclasses with a REMAP ACTIVITY action, the coord_act_exec_time_avg mean of service subclasses an activity is mapped to but does not complete in is unaffected.

Table 312. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wcstats	-
Statistics	event_wlstats	-

Usage

Use this statistic to determine the arithmetic mean of execution time for coordinator activities associated with a service subclass, workload, or work class that completed or aborted.

This average can also be used to determine whether or not the histogram template used for the activity execution time histogram is appropriate. Compute the average activity execution time from the activity execution time histogram. Compare the computed average with this monitor element. If the computed average deviates from the true average reported by this monitor element, consider altering the histogram template for the activity execution time histogram, using a set of bin values that are more appropriate for your data.

coord_act_interarrival_time_avg - Coordinator activity arrival time average monitor element

Arithmetic mean of the time between arrivals of coordinator activities at nesting level 0 associated with this service subclass or work class since the last reset. If the internally tracked average has overflowed, the value -2 is returned. For service subclasses, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service subclass is set to NONE or BASE. For work classes, this monitor elements returns -1 if no COLLECT AGGREGATE ACTIVITY DATA EXTENDED work action is specified for the work class. For workloads, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the workload is set to NONE or BASE. Units are milliseconds.

For service classes, the inter-arrival time mean is calculated for service subclasses through which activities enter the system. When you remap activities between service subclasses with a REMAP ACTIVITY action, the coord_act_interarrival_time_avg of the service subclass you remap an activity to is unaffected.

Table 313. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wcstats	-
Statistics	event_wlstats	-

Usage

Use this statistic to determine the arithmetic mean between arrivals of coordinator activities at nesting level 0 associated with this service subclass, workload, or work class.

The inter-arrival time can be used to determine arrival rate, which is the inverse of inter-arrival time. This average can also be used to determine whether or not the histogram template used for the activity inter-arrival time histogram is appropriate. Compute the average activity inter-arrival time from the activity inter-arrival time histogram. Compare the computed average with this monitor element. If the computed average deviates from the true average reported by this monitor element, consider altering the histogram template for the activity inter-arrival time histogram, using a set of bin values that are more appropriate for your data.

coord_act_lifetime_avg - Coordinator activity lifetime average monitor element

Arithmetic mean of lifetime for coordinator activities at nesting level 0 associated with this service subclass, workload, or work class since the last reset. If the internally tracked average has overflowed, the value -2 is returned. For service subclasses, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service subclass is set to NONE. For work classes, this monitor elements returns -1 if no COLLECT AGGREGATE ACTIVITY DATA work action is specified for the work class. For workloads, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the workload is set to NONE. Units are milliseconds.

For service classes, when you remap activities between service subclasses with a REMAP ACTIVITY action, only the the coord_act_lifetime_avg mean of the final service class where the activity completes is affected.

Table 314. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wcstats	-
Statistics	event_wlstats	-

Usage

Use this statistic to determine the arithmetic mean of the lifetime for coordinator activities associated with a service subclass, workload, or work class that completed or aborted.

This statistic can also be used to determine whether or not the histogram template used for the activity lifetime histogram is appropriate. Compute the average activity lifetime from the activity lifetime histogram. Compare the computed average with this monitor element. If the computed average deviates from the true average reported by this monitor element, consider altering the histogram template for the activity lifetime histogram, using a set of bin values that are more appropriate for your data.

coord_act_lifetime_top - Coordinator activity lifetime top monitor element

High watermark for coordinator activity lifetime, counted at all nesting levels. Units are milliseconds. For service classes, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service class is set to NONE. For work classes, this monitor element returns -1 if no COLLECT AGGREGATE ACTIVITY DATA work action is specified for the work class. For workloads, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the workload is set to NONE.

To effectively use this statistic with service classes when you also remap activities between service subclasses with a REMAP ACTIVITY action, you must aggregate the coord_act_lifetime_top high watermark of any given service subclass with that of other subclasses affected by the same remapping threshold or thresholds. This is because an activity will complete after it has been remapped to a different service

subclass by a remapping threshold, and the time the activity spends in other service subclasses before being remapped is counted only towards the service class in which it completes.

Table 315. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wcstats	-
Statistics	event_scstats	-
Statistics	event_wlstats	-

Usage

This element can be used to help determine whether or not thresholds on activity lifetime are being effective and can also help to determine how to configure such thresholds.

coord_act_queue_time_avg - Coordinator activity queue time average monitor element

Arithmetic mean of queue time for coordinator activities at nesting level 0 associated with this service subclass or work class since the last reset. If the internally tracked average has overflowed, the value -2 is returned. For service subclasses, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service subclass is set to NONE. For work classes, this monitor elements returns -1 if no COLLECT AGGREGATE ACTIVITY DATA work action is specified for the work class. For workloads, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the workload is set to NONE. Units are milliseconds.

For service classes, the queue time counts only towards the service subclass in which the activity completes or is aborted. When you remap activities between service subclasses with a REMAP ACTIVITY action, the coord_act_queue_time_avg mean of service subclasses an activity is mapped to but does not complete in is unaffected.

Table 316. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wcstats	-
Statistics	event_wlstats	-

Usage

Use this statistic to determine the arithmetic mean of the queue time for coordinator activities associated with a service subclass, workload, or work class that completed or aborted.

This statistic can also be used to determine whether or not the histogram template used for the activity queue time histogram is appropriate. Compute the average activity queue time from the activity queue time histogram. Compare the computed average with this monitor element. If the computed average deviates from the true average reported by this monitor element, consider altering the

histogram template for the activity queue time histogram, using a set of bin values that are more appropriate for your data.

coord_act_rejected_total - Coordinator activities rejected total monitor element

The total number of coordinator activities at any nesting level that were rejected instead of being allowed to execute since the last reset. This counter is updated when an activity is prevented from executing by either a predictive threshold or a prevent execution work action. For service classes, the value is updated when the activity completes. For workloads, the value is updated by each workload occurrence at the end of its unit of work.

Table 317. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wlstats	-

Usage

This element can be used to help determine whether or not predictive thresholds and work actions that prevent execution are being effective and whether or not they are too restrictive.

coord_agent_pid - Coordinator agent identifier monitor element

The engine dispatchable unit (EDU) identifier of the coordinator agent for the application. Except on the Linux operating system, the EDU ID is mapped to the thread ID. On the Linux operating system, the EDU ID is a DB2 generated unique identifier.

Table 318. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic

Table 319. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-

Usage

You can use this element to link database system monitor information to other sources of diagnostic information, such as system traces.

coord_agents_top - Maximum Number of Coordinating Agents

The maximum number of coordinating agents working at one time.

Table 320. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

Table 320. Snapshot Monitoring Information (continued)

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Usage

If the peak number of coordinating agents represents too high a workload for this node, you can reduce this upper boundary by changing the **max_coordagents** configuration parameter.

coord_member - Coordinator member monitor element

Coordinating member for the given unit of work or workload.

Table 321. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	Always collected
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics	Always collected
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 322. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-

coord_node - Coordinating Node

In a multi-node system, the node number of the node where the application connected or attached to the instance.

Table 323. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

Table 324. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

Usage Each connected application is served by one coordinator node.

coord_partition_num - Coordinator partition number monitor element

The coordinator partition of the unit of work or activity. In a multi-partition system, the coordinator partition is the partition where the application connected to the database.

Table 325. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-
Activities	event_activity	-
Threshold violations	event_thresholdviolations	-

Usage

This element allows the coordinator partition to be identified for activities or units of work that have records on partitions other than the coordinator.

coord_stmt_exec_time - Execution time for statement by coordinator agent monitor element

The total time spent executing this statement by coordinator agents on this member. The value is given in milliseconds.

Table 326. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 327. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE

corr_token - DRDA Correlation Token

The DRDA AS correlation token.

Table 328. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
Application	appl	Basic

Table 329. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connections	event_connheader	-

Usage The DRDA correlation token is used for correlating the processing between the application server and the application requester. It is an identifier dumped into logs when errors arise, that you can use to identify the conversation that is in error. In some cases, it will be the LUWID of the conversation.

If communications are not using DRDA, this element returns the *appl_id* (see appl_id).

If you are using the database system monitor APIs, note that the API constant SQLM_APPLID_SZ is used to define the length of this element.

cost_estimate_top - Cost estimate top monitor element

The high watermark for the estimated cost of DML activities at all nesting levels in a service subclass or work class. For service subclasses, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service subclass is set to NONE. For work classes, this monitor elements returns -1 if no COLLECT AGGREGATE ACTIVITY DATA work action is specified for the work class.

For service classes, the estimated cost of DML activities is counted only towards the service subclass in which the activity enters the system. When you remap activities between service subclasses with a REMAP ACTIVITY action, the cost_estimate_top of the service subclass you remap an activity to is unaffected.

Table 330. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wcstats	-
Statistics	event_wlstats	-

Usage

Use this element to determine the highest DML activity estimated cost reached on a partition for a service class, workload, or work class in the time interval collected.

count - Number of Event Monitor Overflows

The number of consecutive overflows that have occurred.

Element identifier
count

Element type
counter

Table 331. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Overflow Record	event_overflow	-

Usage You may use this element to get an indication of how much monitor data has been lost.

The event monitor sends one overflow record for a set of consecutive overflows.

cpu_configured - Number of configured CPUs monitor element

The number of processors on this host that the operating system is aware of.

Table 332. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

cpu_cores_per_socket - Number of CPU cores per socket monitor element

The number of processors on this host. On single core systems, this value is 1.

Table 333. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

cpu_hmt_degree - Number of logical CPUs monitor element

On systems that support hardware multithreading, the number of logical processors that appear to be present as a result of multithreading. On systems that do not support multithreading, this value is 1.

Table 334. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

cpu_idle - Processor idle time monitor element

Processor idle time, expressed in processor ticks. Reported for Windows, AIX and Linux systems only. This measurement represents the aggregate for all processors on the system.

Table 335. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

Usage

- This measurement represents the aggregate for all processors on the system.
- On AIX, this metric is reported for the workload partition (WPAR) and the logical partition (LPAR) on which the DB2 server is running.
- You can use this monitor element along with the related processor timer elements to calculate processor utilization for a specific time interval on the host system. To calculate processor utilization as a percentage, perform the following steps:
  1. Use the ENV_GET_SYSTEM_RESOURCES function at the beginning to the time interval to retrieve the values for the following metrics:
     - cpu_user_{t1} = **cpu_user**
     - cpu_system_{t1} = **cpu_system**
     - cpu_idle_{t1} = **cpu_idle**
     - cpu_wait_{t1} = **cpu_wait**
  2. Repeat the preceding step to determine the timestamps for the same metrics at the end of the time interval for which you want to calculate processor utilization:
     - cpu_user_{t2} = **cpu_user**
     - cpu_system_{t2} = **cpu_system**
     - cpu_idle_{t2} = **cpu_idle**
     - cpu_iowait_{t2} = **cpu_iowait**
  3. Calculate processor utilization using the following formula:

$$100 \times \frac{(cpu_system_{t2} - cpu_system_{t1}) + (cpu_user_{t2} - cpu_user_{t1})}{(cpu_system_{t2} - cpu_system_{t1}) + (cpu_user_{t2} - cpu_user_{t1}) + (cpu_idle_{t2} - cpu_idle_{t1}) + (cpu_iowait_{t2} - cpu_iowait_{t1})}$$

cpu_iowait - IO Wait time monitor element

Time spent waiting for IO (Linux, UNIX); time spent receiving and servicing hardware interrupts (Windows), expressed in processor ticks. Reported for Windows, AIX and Linux systems only. This measurement represents the aggregate for all processors on the system.

Table 336. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

Usage

- This measurement represents the aggregate for all processors on the system.
- On AIX, this metric is reported for the workload partition (WPAR) and the logical partition (LPAR) on which the DB2 server is running.
- You can use this monitor element along with the related processor timer elements to calculate processor utilization for a specific time interval on the host system. To calculate processor utilization as a percentage, perform the following steps:
  1. Use the ENV_GET_SYSTEM_RESOURCES function at the beginning to the time interval to retrieve the values for the following metrics:
     - cpu_user_{t1} = **cpu_user**

- cpu_system_{t1} = **cpu_system**
- cpu_idle_{t1} = **cpu_idle**
- cpu_wait_{t1} = **cpu_wait**

2. Repeat the preceding step to determine the timestamps for the same metrics at the end of the time interval for which you want to calculate processor utilization:
   - cpu_user_{t2} = **cpu_user**
   - cpu_system_{t2} = **cpu_system**
   - cpu_idle_{t2} = **cpu_idle**
   - cpu_iowait_{t2} = **cpu_iowait**
3. Calculate processor utilization using the following formula:

$$100 \times \frac{(cpu_system_{t2} - cpu_system_{t1}) + (cpu_user_{t2} - cpu_user_{t1})}{(cpu_system_{t2} - cpu_system_{t1}) + (cpu_user_{t2} - cpu_user_{t1}) + (cpu_idle_{t2} - cpu_idle_{t1}) + (cpu_iowait_{t2} - cpu_iowait_{t1})}$$

cpu_load_long - Processor load (long timeframe) monitor element

Processor load over the longer term, as defined by the system. For example, the average processor load over the past 10 or 15 minutes. Reported for all platforms except Windows.

Table 337. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

cpu_load_medium - Processor load (medium timeframe) monitor element

Processor load over the medium term, as defined by the system. For example, the average processor load over the past 5 or 10 minutes. Reported for all platforms except Windows.

Table 338. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

cpu_load_short - Processor load (short timeframe) monitor element

Processor load over the short term, as defined by the system. For example, the average processor load over the past 1 or 5 minutes. Reported for all platforms except Windows.

Table 339. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

cpu_online - Number of CPUs online monitor element

The number of processors on this host that are currently online.

Table 340. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

cpu_speed - CPU clock speed monitor element

The clock speed of the processors on this host, in MHz.

Table 341. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

cpu_system - Kernel time monitor element

Time spent running kernel code, expressed in processor ticks. Reported for Windows, AIX and Linux systems only. This measurement represents the aggregate for all processors on the system.

Table 342. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected
ENV_GET_DB2_SYSTEM_RESOURCES table function - Return DB2(r) system information	Always collected

Usage

- This measurement represents the aggregate for all processors on the system.
- On AIX, this metric is reported for the workload partition (WPAR) and the logical partition (LPAR) on which the DB2 server is running.
- You can use this monitor element along with the related processor timer elements to calculate processor utilization for a specific time interval on the host system. To calculate processor utilization as a percentage, perform the following steps:
  1. Use the ENV_GET_SYSTEM_RESOURCES function at the beginning to the time interval to retrieve the values for the following metrics:
     - cpu_user_{t1} = **cpu_user**
     - cpu_system_{t1} = **cpu_system**
     - cpu_idle_{t1} = **cpu_idle**
     - cpu_wait_{t1} = **cpu_wait**
  2. Repeat the preceding step to determine the timestamps for the same metrics at the end of the time interval for which you want to calculate processor utilization:
     - cpu_user_{t2} = **cpu_user**
     - cpu_system_{t2} = **cpu_system**

- cpu_idle_{t2} = **cpu_idle**
- cpu_iowait_{t2} = **cpu_iowait**

3. Calculate processor utilization using the following formula:

$$100 \times \frac{(cpu_system_{t2} - cpu_system_{t1}) + (cpu_user_{t2} - cpu_user_{t1})}{(cpu_system_{t2} - cpu_system_{t1}) + (cpu_user_{t2} - cpu_user_{t1}) + (cpu_idle_{t2} - cpu_idle_{t1}) + (cpu_iowait_{t2} - cpu_iowait_{t1})}$$

cpu_timebase - Frequency of timebase register increment monitor element

The frequency, in Hz, at which the timebase register is incremented. For Linux and PowerPC® systems only

Table 343. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

cpu_total - Number of CPUs monitor element

The number of processors on this host.

Table 344. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

Usage

The number reported for this monitor element means different things in different operating environments. For example, when returned from a Windows, system, **cpu_total** refers to the total number of processors installed; on AIX, it represents the number of configured processors.

cpu_usage_total - Processor usage monitor element

The overall processor usage on this host including kernel processing time, expressed as a percentage. Reported for AIX, Linux and Windows systems only.

Table 345. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

cpu_user - Non-kernel processing time monitor element

Time spent running user (non-kernel) code, expressed in processor ticks. Reported for Windows, AIX, and Linux systems only. This measurement represents the aggregate for all processors on the system.

Table 346. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected
ENV_GET_DB2_SYSTEM_RESOURCES table function - Return DB2(r) system information	Always collected

Usage

- This measurement represents the aggregate for all processors on the system.
- On AIX, this metric is reported for the workload partition (WPAR) and the logical partition (LPAR) on which the DB2 server is running.
- You can use this monitor element along with the related processor timer elements to calculate processor utilization for a specific time interval on the host system. To calculate processor utilization as a percentage, perform the following steps:
  1. Use the ENV_GET_SYSTEM_RESOURCES function at the beginning to the time interval to retrieve the values for the following metrics:
     - cpu_user_{t1} = **cpu_user**
     - cpu_system_{t1} = **cpu_system**
     - cpu_idle_{t1} = **cpu_idle**
     - cpu_wait_{t1} = **cpu_wait**
  2. Repeat the preceding step to determine the timestamps for the same metrics at the end of the time interval for which you want to calculate processor utilization:
     - cpu_user_{t2} = **cpu_user**
     - cpu_system_{t2} = **cpu_system**
     - cpu_idle_{t2} = **cpu_idle**
     - cpu_iowait_{t2} = **cpu_iowait**
  3. Calculate processor utilization using the following formula:

$$100 \times \frac{(cpu_system_{t2} - cpu_system_{t1}) + (cpu_user_{t2} - cpu_user_{t1})}{(cpu_system_{t2} - cpu_system_{t1}) + (cpu_user_{t2} - cpu_user_{t1}) + (cpu_idle_{t2} - cpu_idle_{t1}) + (cpu_iowait_{t2} - cpu_iowait_{t1})}$$

cputime_threshold_id - CPU time threshold ID monitor element

The ID of the CPUTIME threshold that was applied to the activity.

Table 347. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand which CPUTIME threshold, if any, was applied to the activity.

cputime_threshold_value - CPU time threshold value monitor element

The upper bound of the CPUTIME threshold that was applied to the activity.

Table 348. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand the value of the CPUTIME threshold applied to the activity, if any.

cputime_threshold_violated - CPU time threshold violated monitor element

This monitor element returns 'Yes' to indicate that the activity violated the CPUTIME threshold. 'No' indicates that the activity has not yet violated the threshold.

Table 349. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to determine if the activity violated the CPUTIME threshold that was applied to the activity.

cputimeinsc_threshold_id - CPU time in service class threshold ID monitor element

The ID of the CPUTIMEINSC threshold that was applied to the activity.

Table 350. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand which CPUTIMEINSC threshold, if any, was applied to the activity.

cputimeinsc_threshold_value - CPU time in service class threshold value monitor element

The upper bound of the CPUTIMEINSC threshold that was applied to the activity.

Table 351. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand the value of the CPUTIMEINSC threshold applied to the activity, if any.

cputimeinsc_threshold_violated - CPU time in service class threshold violated monitor element

This monitor element returns 'Yes' to indicate that the activity violated the CPUTIMEINSC threshold. 'No' indicates that the activity has not yet violated the threshold.

Table 352. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to determine if the activity violated the CPUTIMEINSC threshold that was applied to the activity.

create_nickname - Create Nicknames

This element contains a count of the total number of times the federated server has created a nickname over an object residing on this data source on behalf of any application since the start of the federated server instance or the last reset of the database monitor counters, whichever is latest.

Table 353. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

Usage

Use this element to determine the amount of CREATE NICKNAME activity against this data source by this federated server instance or an application. CREATE NICKNAME processing results in multiple queries running against the data source catalogs; therefore, if the value of this element is high, you should determine the cause and perhaps restrict this activity.

create_nickname_time - Create Nickname Response Time

This element contains the aggregate amount of time, in milliseconds, that it has taken this data source to process CREATE NICKNAME statements from all applications or a single application running on this federated server instance. The response time is measured since the start of the federated server instance, or the last reset of the database monitor counter, whichever is the latest. The response time is measured as the difference between the time the federated server started retrieving information from the data source to process the CREATE NICKNAME statement, and the time it took to retrieve all the required data from the data source.

Table 354. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Timestamp
Application	appl_remote	Timestamp

For snapshot monitoring, this counter can be reset.

Usage Use this element to determine how much actual time was used to create nicknames for this data source.

creator - Application Creator

The authorization ID of the user that pre-compiled the application.

Table 355. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement
DCS Statement	dcs_stmt	Statement

Table 356. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	event_dlconn	-
Statements	event_stmt	-
Activities	event_activitystmt	-

Usage Use this element to help identify the SQL statement that is processing, in conjunction with the CREATOR column of the package section information in the catalogs.

If the CURRENT PACKAGE PATH special register is set, the *creator* value may reflect different values over the lifetime of the SQL statement. If a

snapshot or event monitor record is taken before PACKAGE PATH resolution, the *creator* value will reflect the value flowed in from the client request. If a snapshot or event monitor record is taken after PACKAGE PATH resolution, the *creator* value will reflect the creator of the resolved package. The resolved package will be the package whose *creator* value appears earliest in the CURRENT PACKAGE PATH SPECIAL REGISTER and whose package name and unique ID matches that of the client request.

current_active_log - Current Active Log File Number

The file number of the active log file the DB2 database system is currently writing.

Table 357. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	detail_log	Basic

Table 358. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage Use this element in conjunction with the *first_active_log* and *last_active_log* elements to determine the range of active log files. Knowing the range of active log files helps you determine the disk space required for log files.

You can also use this element to determine which log files have data to help you identify log files needed for split mirror support.

current_archive_log - Current Archive Log File Number

The file number of the log file the DB2 database system is currently archiving. If the DB2 database system is not archiving a log file, the value for this element is SQLM_LOGFILE_NUM_UNKNOWN.

Table 359. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	detail_log	Basic

Table 360. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage Use this element to determine if there is a problem archiving log files. Such problems include:

- Slow archive media
- Archive media that is not available

current_extent - Extent currently being moved monitor element

The numeric identifier of the extent currently being moved by the table space rebalancing process.

Table 361. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_EXTENT_MOVEMENT_STATUS - Get extent movement progress status metrics	Always collected

cursor_name - Cursor Name

The name of the cursor corresponding to this SQL statement.

Element identifier
cursor_name

Element type
information

Table 362. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement

Table 363. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks with Details	event_detailed_dlconn	-
Statements	event_stmt	-

Usage You may use this element to identify the SQL statement that is processing. This name will be used on an OPEN, FETCH, CLOSE, and PREPARE of an SQL SELECT statement. If a cursor is not used, this field will be blank.

data_object_pages - Data Object Pages

The number of disk pages consumed by a table. This size represents the base table size only. Space consumed by index objects, LOB data, and long data are reported by *index_object_pages*, *lob_object_pages*, and *long_object_pages*, respectively.

Table 364. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table	Basic

Table 365. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Tables	event_table	-

Usage This element provides a mechanism for viewing the actual amount of

space consumed by a particular table. This element can be used in conjunction with a table event monitor to track the rate of table growth over time.

data_object_l_pages - Table data logical pages monitor element

The number of logical pages used on disk by data contained in this table.

Table 366. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLE table function - Get table metrics	DATA OBJECT METRICS BASE

Usage

- This value might be less than the amount of space allocated for the object. This can happen when you use the RECLAIM EXTENTS ONLY option with the REORG TABLE command. In this case, reclaimed extents are included in the logical number of pages returned by MON_GET_TABLE.
- This value might be less than the amount of space physically allocated for the object. This can happen when you use the REUSE STORAGE option of the TRUNCATE statement. This option causes storage allocated for the table to continue to be allocated, although the storage will be considered empty. In addition, the value for this monitor element might be less than the amount of space logically allocated for the object, because the total space logically allocated includes a small amount of additional meta data.

  To retrieve an accurate measure of the logical or physical size of an object, use the ADMIN_GET_TAB_INFO_V97 function. This function provides more accurate information about the size of objects than you can obtain by multiplying the number of pages reported for this monitor element by the page size.

data_partition_id - Data partition identifier monitor element

The identifier of the data partition for which information is returned.

Table 367. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLE table function - Get table metrics	Always collected
MON_GET_INDEX table function - Get index metrics	Always collected

Table 368. Snapshot monitoring information

Snapshot level	Logical data grouping	Monitor switch
Table	table	Basic
Lock	lock	Lock
Lock	lock_wait	Lock

Table 369. Event monitoring information

Event type	Logical data grouping	Monitor switch
Table	event_table	-
Deadlocks	event_dlconn	-
Deadlocks with Details	event_detailed_dlconn	-
Deadlocks	lock	-

Usage

This element is only applicable to partitioned tables and partitioned indexes. Otherwise, the value of this monitor element is NULL.

When returning lock level information, a value of -1 represents a lock which controls access to the whole table.

datasource_name - Data Source Name

This element contains the name of the data source whose remote access information is being displayed by the federated server. This element corresponds to the 'SERVER' column in SYSCAT.SERVERS.

Element identifier
datasource_name

Element type
information

Table 370. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Basic
Application	appl_remote	Basic

Usage Use this element to identify the data source whose access information has been collected and is being returned.

db2_process_id - DB2 process ID monitor element

Numeric identifier of the DB2 process running on the reported member.

Table 371. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_DB2_SYSTEM_RESOURCES table function - Return DB2(r) system information	Always collected

db2_process_name - DB2 process name monitor element

Name of the DB2 process running on the reported member.

Table 372. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_DB2_SYSTEM_RESOURCES table function - Return DB2(r) system information	Always collected

db2_status - Status of DB2 Instance

The current status of the instance of the database manager.

Table 373. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

Usage You can use this element to determine the state of your database manager instance.

Values for this element are:

API Constant	Value	Description
SQLM_DB2_ACTIVE	0	The database manager instance is active.
SQLM_DB2_QUIESCE_PEND	1	The instance and the databases in the instance are in quiesce-pending state. New connections to any instance database are not permitted and new units of work cannot be started. Depending on the quiesce request, active units of work will be allowed to complete or will be rolled back immediately.
SQLM_DB2_QUIESCED	2	The instance and the databases in the instance has been quiesced. New connections to any instance database are not permitted and new units of work cannot be started.

db2start_time - Start Database Manager Timestamp

The date and time that the database manager was started using the `db2start` command.

Element identifier
db2start_time

Element type
timestamp

Table 374. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

Usage This element may be used with the *time_stamp* monitor element to calculate the elapsed time since the database manager was started up until the snapshot was taken.

db_conn_time - Database activation timestamp monitor element

The date and time of the connection to the database (at the database level, this is the first connection to the database), or when the activate database was issued.

Table 375. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Timestamp
Table Space	tablespace_list	Buffer Pool, Timestamp
Table	table_list	Timestamp

Table 376. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-

Usage

Use this element with the **disconn_time** monitor element to calculate the total connection time.

db_heap_top - Maximum Database Heap Allocated

This element is being maintained for DB2 version compatibility. It now measures memory usage, but not exclusively usage by the database heap.

Note: The **db_heap_top** monitor element is deprecated starting with DB2 Version 9.5. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

Table 377. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Table 378. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

db_location - Database Location

The location of the database in relation to the application.

Table 379. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Usage Determine the relative location of the database server with respect to the application taking the snapshot. Values are:

- SQLM_LOCAL
- SQLM_REMOTE

db_name - Database Name

The real name of the database for which information is collected or to which the application is connected. This is the name the database was given when created.

Table 380. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_MEMORY_POOL table function - Get memory pool information	Always collected
MON_GET_MEMORY_SET table function - Get memory set information	Always collected

Table 381. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Database	dbase_remote	Basic
Application	appl_id_info	Basic
Application	appl_remote	Basic
Table Space	tablespace_list	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Table	table_list	Table
Lock	db_lock_list	Basic
Dynamic SQL	dynsql_list	Basic
DCS Database	dcs_dbase	Basic
DCS Application	dcs_appl_info	Basic

Table 382. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_dbheader	-

Usage You may use this element to identify the specific database to which the data applies.

For applications that are not using DB2 Connect to connect to a host or System i® database server, you can use this element in conjunction with the **db_path** monitor element to uniquely identify the database and help relate the different levels of information provided by the monitor.

When returned with information about memory pools with MON_GET_MEMORY_POOL, the value for this monitor element is NULL for memory pools defined at the instance level. Otherwise, the value returned is the name of the database for which the pool is defined.

db_path - Database Path

The full path of the location where the database is stored on the monitored system.

Table 383. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl_id_info	Basic
Table Space	tablespace_list	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Table	table_list	Table
Lock	db_lock_list	Basic
Dynamic SQL	dynsql_list	Basic

Table 384. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_dbheader	-

Usage This element can be used with the *db_name* monitor element to identify the specific database to which the data applies.

db_status - Status of Database

The current status of the database.

Table 385. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Usage You can use this element to determine the state of your database.

Values for this field are:

API Constant	Value	Description
SQLM_DB_ACTIVE	0	The database is active.
SQLM_DB_QUIESCE_PEND	1	The database is in quiesce-pending state. New connections to the database are not permitted and new units of work cannot be started. Depending on the quiesce request, active units of work will be allowed to complete or will be rolled back immediately.
SQLM_DB_QUIESCED	2	The database has been quiesced. New connections to the database are **not** permitted and new units of work cannot be started.
SQLM_DB_ROLLFWD	3	A rollforward is in progress on the database.
SQLM_DB_ACTIVE_STANDBY	4	The database is a read-enabled HADR standby database.

API Constant	Value	Description
SQLM_DB_STANDBY	5	The database is an HADR standby database.

db_storage_path - Automatic storage path monitor element

This element shows the full path of a location that is used by the database for placing automatic storage table spaces. There can be 0 or more storage paths associated with a database.

Table 386. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	db_sto_path_info	Basic

Usage

Use this element with the **num_db_storage_paths** monitor element to identify the storage paths that are associated with this database.

db_storage_path_state - Storage path state monitor element

The automatic storage path state indicates whether the storage path is in use by the database.

Table 387. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	db_sto_path_info	Basic

Usage

Use this monitor element to determine whether the storage path is in use by the database. The following values are possible:

NOT_IN_USE
There are no table spaces using this storage path on the specified database partition.

IN_USE
There are table spaces using this storage path on the specified database partition.

DROP_PENDING
This storage path has been dropped, but some table spaces are still using it. Before storage paths are physically dropped from the database, all table spaces must stop using them. To stop using a dropped storage path, either drop the table space or rebalance the table space using the REBALANCE clause of the ALTER TABLESPACE statement.

db_storage_path_with_dpe - Storage path including database partition expression monitor element

Automatic storage path that includes the unevaluated database partition expression.

Table 388. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	db_sto_path_info	Basic

Usage

Use this monitor element to determine the storage path that was specified for the database as part of the CREATE DATABASE command or the ALTER DATABASE statement, if the storage path contains a database partition expression.

If the storage path does not contain a database partition expression, then this monitor element returns a null value.

db_work_action_set_id - Database work action set ID monitor element

If this activity has been categorized into a work class of database scope, this monitor element shows the ID of the work action set associated with the work class set to which the work class belongs. Otherwise, this monitor element shows the value of 0.

Table 389. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
WLM_GET_ACTIVITY_DETAILS_COMPLETE (reported in DETAILS XML document)	Always collected

Table 390. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

Usage

This element can be used with the **db_work_class_id** element to uniquely identify the database work class of the activity, if one exists.

db_work_class_id - Database work class ID monitor element

If this activity has been categorized into a work class of database scope, this monitor element displays the ID of the work class. Otherwise, this monitor element displays the value of 0.

Table 391. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
WLM_GET_ACTIVITY_DETAILS_COMPLETE table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 392. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

Usage

This element can be used with the **db_work_action_set_id** element to uniquely identify the database work class of the activity, if one exists.

dcs_appl_status - DCS Application Status

The status of a DCS application at the DB2 Connect gateway.

Table 393. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcs_appl_info	Basic

Usage Use this element for problem determination on DCS applications. Values are:

- SQLM_DCS_CONNECTPEND_OUTBOUND

  The application has initiated a database connection from the DB2 Connect gateway to the host database, but the request has not completed yet.
- SQLM_DCS_UOWWAIT_OUTBOUND

  The DB2 Connect gateway is waiting for the host database to reply to the application's request.
- SQLM_DCS_UOWWAIT_INBOUND

  The connection from the DB2 Connect gateway to the host database has been established and the gateway is waiting for SQL requests from the application. Or the DB2 Connect gateway is waiting on behalf of the unit of work in the application. This usually means that the application's code is being executed.

dcs_db_name - DCS Database Name

The name of the DCS database as cataloged in the DCS directory.

Table 394. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Basic
DCS Application	dcs_appl_info	Basic

Usage Use this element for problem determination on DCS applications.

ddl_sql_stmts - Data Definition Language (DDL) SQL Statements

This element indicates the number of SQL Data Definition Language (DDL) statements that were executed.

Table 395. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 396. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage You can use this element to determine the level of database activity at the application or database level. DDL statements are expensive to run due to their impact on the system catalog tables. As a result, if the value of this element is high, you should determine the cause, and possibly restrict this activity from being performed.

You can also use this element to determine the percentage of DDL activity using the following formula:

```
ddl_sql_stmts / total number of statements
```

This information can be useful for analyzing application activity and throughput. DDL statements can also impact:

- the catalog cache, by invalidating table descriptor information and authorization information that are stored there and causing additional system overhead to retrieve the information from the system catalogs
- the package cache, by invalidating sections that are stored there and causing additional system overhead due to section recompilation.

Examples of DDL statements are CREATE TABLE, CREATE VIEW, ALTER TABLE, and DROP INDEX.

deadlock_id - Deadlock Event Identifier

The deadlock identifier for a deadlock.

Element identifier
deadlock_id

Element type
information

Table 397. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	event_deadlock	-
Deadlocks	event_dlconn	-
Deadlocks with Details	event_detailed_dlconn	-

Table 397. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks with Details History	event_detailed_dlconn	-
Deadlocks with Details History	event_stmt_history	-
Deadlocks with Details History Values	event_data_value	-
Deadlocks with Details History Values	event_detailed_dlconn	-
Deadlocks with Details History Values	event_stmt_history	-

Usage Use this element in your monitoring application to correlate deadlock connection and statement history event records with deadlock event records.

deadlock_node - Partition Number Where Deadlock Occurred

Partition number where the deadlock occurred.

Element identifier
deadlock_node

Element type
information

Table 398. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	event_deadlock	-
Deadlocks	event_dlconn	-
Deadlocks with Details	event_detailed_dlconn	-

Usage This element is relevant only for partitioned databases. Use this in your monitoring application to correlate deadlock connection event records with deadlock event records.

deadlocks - Deadlocks detected monitor element

The total number of deadlocks that have occurred.

Table 399. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 399. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 400. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Lock

For snapshot monitoring, this counter can be reset.

Table 401. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Database	event_db	-
Connection	event_conn	-
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE

Table 401. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

This element can indicate that applications are experiencing contention problems. These problems could be caused by the following situations:

- Lock escalations are occurring for the database
- An application may be locking tables explicitly when system-generated row locks may be sufficient
- An application may be using an inappropriate isolation level when binding
- Catalog tables are locked for repeatable read
- Applications are getting the same locks in different orders, resulting in deadlock.

You may be able to resolve the problem by determining in which applications (or application processes) the deadlocks are occurring. You may then be able to modify the application to better enable it to run concurrently. Some applications, however, may not be capable of running concurrently.

You can use the connection timestamp monitor elements (**last_reset**, **db_conn_time**, and **appl_con_time**) to determine the severity of the deadlocks. For example, 10 deadlocks in 5 minutes is much more severe than 10 deadlocks in 5 hours.

The descriptions for the related elements listed above may also provide additional tuning suggestions.

degree_parallelism - Degree of Parallelism

The degree of parallelism requested when the query was bound.

Element identifier
degree_parallelism

Element type
information

Table 402. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement

Usage Use with agents_top, to determine if the query achieved maximum level of parallelism.

del_keys_cleaned - Pseudo deleted keys cleaned monitor element

Number of pseudo deleted keys that have been cleaned.

Table 403. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

delete_sql_stmts - Deletes

This element contains a count of the total number of times the federated server has issued a DELETE statement to this data source on behalf of any application since the start of the federated server instance or the last reset of the database monitor counters, whichever is latest.

Table 404. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

Usage Use this element to determine the level of database activity directed against this data source by the federated server or an application.

You can also use this element to determine the percentage of write activity against this data source by the federated server or an application, with the following formula:

```
write_activity =
  (INSERT statements + UPDATE statements + DELETE statements ) /
  (SELECT statements + INSERT statements + UPDATE statements +
  DELETE statements)
```

delete_time - Delete Response Time

This element contains the aggregate amount of time, in milliseconds, that it has taken this data source to respond to DELETEs from all applications or a single application running on this federated server instance since the start of the federated server instance or the last reset of the database monitor counters, whichever is latest.

The response time is measured as the difference in time between the time the federated server submits a DELETE statement to the data source, and the time the data source responds to the federated server, indicating the DELETE has been processed.

Table 405. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Timestamp
Application	appl_remote	Timestamp

For snapshot monitoring, this counter can be reset.

Usage Use this element to determine how much actual time transpires while waiting for DELETEs to this data source to be processed. This information can be useful for capacity planning and tuning.

destination_service_class_id – Destination service class ID monitor element

The ID of the service subclass to which an activity was remapped when the threshold violation record to which this element belongs was generated. This element has a value of zero for any threshold action other than REMAP ACTIVITY.

Table 406. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Threshold violations	event_thresholdviolations	-

Usage

Use this element to trace the path of an activity through the service classes to which it was remapped. This element can also be used to compute aggregates of how many activities were mapped into a given service subclass.

diaglog_write_wait_time - Diagnostic log file write wait time monitor element

The time spent waiting on a write to the `db2diag` log file. The value is given in milliseconds.

Table 407. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 408. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element to understand the amount of time spent writing to the `db2diag` log file. In a partitioned database environment, a high value for this time may indicate contention for the `db2diag` log file if shared storage is being used for the diagnostic directory path (diagpath). A high value may also indicate excessive logging, for example if **`diaglevel`** has been set to log all informational messages.

diaglog_writes_total - Total diagnostic log file writes monitor element

The number of times agents have written to the `db2diag` log file.

Table 409. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

Table 409. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 410. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element with the **diaglog_write_wait_time** monitor element to understand the average amount of time spent writing to the `db2diag` log file.

direct_read_reqs - Direct read requests monitor element

The number of requests to perform a direct read of one or more sectors of data.

Table 411. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 411. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 412. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 413. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Database	event_db	-
Connection	event_conn	-
Tablespaces	event_tablespace	-

Table 413. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use the following formula to calculate the average number of sectors that are read by a direct read:

```
direct_reads / direct_read_reqs
```

direct_read_time - Direct read time monitor element

The elapsed time required to perform the direct reads. This value is given in milliseconds.

Table 414. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE

Table 414. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 415. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 416. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Database	event_db	-
Connection	event_conn	-
Tablespaces	event_tablespace	-
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use the following formula to calculate the average direct read time per sector:
`direct_read_time / direct_reads`

A high average time may indicate an I/O conflict.

direct_reads - Direct reads from database monitor element

The number of read operations that do not use the buffer pool.

Table 417. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 418. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 419. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Database	event_db	-
Connection	event_conn	-
Tablespaces	event_tablespace	-
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use the following formula to calculate the average number of sectors that are read by a direct read:

```
direct_reads / direct_read_reqs
```

When using system monitors to track I/O, this element helps you distinguish database I/O from non-database I/O on the device.

Direct reads are performed in units, the smallest being a 512-byte sector. They are used when:

- Reading LONG VARCHAR columns
- Reading LOB (large object) columns
- Performing a backup

direct_write_reqs - Direct write requests monitor element

The number of requests to perform a direct write of one or more sectors of data.

Table 420. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 421. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 422. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Database	event_db	-
Connection	event_conn	-
Tablespaces	event_tablespace	-
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use the following formula to calculate the average number of sectors that are written by a direct write:

```
direct_writes / direct_write_reqs
```

direct_write_time - Direct write time monitor element

The elapsed time required to perform the direct writes. This value is reported in milliseconds.

Table 423. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 423. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 424. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 425. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE

Table 425. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-
Tablespaces	event_tablespace	-
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use the following formula to calculate the average direct write time per sector:

```
direct_write_time / direct_writes
```

A high average time may indicate an I/O conflict.

direct_writes - Direct writes to database monitor element

The number of write operations that do not use the buffer pool.

Table 426. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 426. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 427. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 428. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Database	event_db	-
Connection	event_conn	-
Tablespaces	event_tablespace	-
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use the following formula to calculate the average number of sectors that are written by a direct write.

```
direct_writes / direct_write_reqs
```

When using system monitors to track I/O, this element helps you distinguish database I/O from non-database I/O on the device.

Direct writes are performed in units, the smallest being a 512-byte sector. They are used when:

- Writing LONG VARCHAR columns
- Writing LOB (large object) columns
- Performing a restore
- Performing a load
- Allocating new extents for SMS table space if MPFA is enabled (which is the default)

disconn_time - Database Deactivation Timestamp

The date and time that the application disconnected from the database (at the database level, this is the time the last application disconnected).

Element identifier
disconn_time

Element type
timestamp

Table 429. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage Use this element to calculate the elapsed time since:

- The database was active (for information at the database level)
- The connection was active (for information at the connection level).

disconnects - Disconnects

This element contains a count of the total number of times the federated server has disconnected from this data source on behalf of any application since the start of the federated server instance or the last reset of the database monitor counters, whichever is latest.

Table 430. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Basic

For snapshot monitoring, this counter can be reset.

Usage

Use this element to determine the total number of times the federated server has disconnected from this data source on behalf of any application. Together with the CONNECT count, this element provides a mechanism by which you can determine the number of applications this instance of the federated server believes is currently connected to a data source.

dl_conns - Connections involved in deadlock monitor element

The number of connections that are involved in the deadlock.

Table 431. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks[1]	event_deadlock	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

Use this element in your monitoring application to identify how many deadlock connection event records will follow in the event monitor data stream.

dynamic_sql_stmts - Dynamic SQL Statements Attempted

The number of dynamic SQL statements that were attempted.

Table 432. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 433. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage You can use this element to calculate the total number of successful SQL statements at the database or application level:

```
  dynamic_sql_stmts
+ static_sql_stmts
- failed_sql_stmts
= throughput during monitoring period
```

edu_ID - Engine dispatchable unit ID monitor element

ID of engine dispatchable unit with which this memory pool is associated.

Table 434. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_MEMORY_POOL table function - Get memory pool infromation	Always collected

Usage

When returned by the table function MON_GET_MEMORY_POOL, this monitor element is NULL except when the memory pool being described is PRIVATE.

eff_stmt_text - Effective statement text monitor element

The effective text of the SQL statement, if the statement was modified as a result of the statement concentrator.

Table 435. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
WLM_GET_ACTIVITY_DETAILS_COMPLETE table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 436. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activity	event_activitystmt	-

Usage

If the statement concentrator is enabled and if the statement text has been modified as a result of the statement concentrator, then this monitor element contains the effective statement text. Otherwise, this monitor element contains a text string which is 0 bytes long.

effective_isolation - Effective isolation monitor element

The effective isolation level for this statement.

Table 437. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	Always collected

Table 438. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Package cache	-	COLLECT BASE DATA

Usage

Use this element to understand the isolation level that was used during the execution of the statement.

effective_lock_timeout - Effective lock timeout monitor element

The effective lock timeout value for this activity. This value is reported in seconds.

Table 439. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

effective_query_degree - Effective query degree monitor element

The effective query degree of parallelism for this activity.

Table 440. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 441. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-

elapsed_exec_time - Statement Execution Elapsed Time

At the DCS statement level, this is the elapsed time spent processing an SQL request on a host database server. This value is reported by this server. For event monitors that write to tables, the value of this element is given in microseconds by using the BIGINT data type. In contrast to the host_response_time element, this element does not include the network elapsed time between DB2 Connect and the host database server. At other levels, this value represents the sum of the host execution times for all the statements that were executed for a particular database or application, or for those statements that used a given number of data transmissions.

Table 442. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Statement, Timestamp
Application	appl	Statement, Timestamp
DCS Database	dcs_dbase	Statement, Timestamp
DCS Application	dcs_appl	Statement, Timestamp
DCS Statement	dcs_stmt	Statement, Timestamp
Data Transmission	stmt_transmissions	Statement, Timestamp

For snapshot monitoring at the statement level, this counter cannot be reset. This counter can be reset at other levels.

Usage Use this element, along with other elapsed time monitor elements, to evaluate the database server's processing of SQL requests and to help isolate performance issues.

Subtract this element from the host_response_time element to calculate the network elapsed time between DB2 Connect and the host database server.

Note: For the dcs_dbase, dcs_appl, dcs_stmt and stmt_transmissions levels, the *elapsed_exec_time element* applies only to z/OS® databases. If the DB2 Connect gateway is connecting to a Windows, Linux, AIX, or other UNIX database, the *elapsed_exec_time* is reported as zero.

empty_pages_deleted - Empty pages deleted monitor element

The number of pseudo empty pages that have been deleted. Pseudo empty pages are pages where all the keys have been pseudo deleted.

Table 443. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

empty_pages_reused - Empty pages reused monitor element

The number of pseudo empty pages that have been reused. Pseudo empty pages are pages where all the keys have been pseudo deleted.

Table 444. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

entry_time - Entry time monitor element

The time at which this activity entered the system.

Table 445. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

estimatedsqlcost_threshold_id - Estimated SQL cost threshold ID monitor element

The ID of the ESTIMATEDSQLCOST threshold that was applied to the activity.

Table 446. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand which ESTIMATEDSQLCOST threshold, if any, was applied to the activity.

estimatedsqlcost_threshold_value - Estimated SQL cost threshold value monitor element

The upper bound of the ESTIMATEDSQLCOST threshold that was applied to the activity.

Table 447. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand the value of the ESTIMATEDSQLCOST threshold applied to the activity, if any.

estimatedsqlcost_threshold_violated - Estimated SQL cost threshold violated monitor element

This monitor element returns 'Yes' to indicate that the activity violated the ESTIMATEDSQLCOST threshold. 'No' indicates that the activity has not yet violated the threshold.

Table 448. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to determine if the activity violated the ESTIMATEDSQLCOST threshold that was applied to the activity.

event_monitor_name - Event Monitor Name

The name of the event monitor that created the event data stream.

Element identifier
event_monitor_name

Element type
information

Table 449. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Event Log Header	event_log_header	-

Usage This element allows you to correlate the data that you are analyzing to a specific event monitor in the system catalog tables. This is the same name that can be found in the NAME column of the SYSCAT.EVENTMONITORS catalog table, which is the name specified on the CREATE EVENT MONITOR and SET EVENT MONITOR statements.

event_time - Event Time

The date and time an event occurred.

Table 450. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Tablespaces	event_tablespace	-
Tables	event_table	-

Usage You can use this element to help relate events chronologically.

evmon_activates - Number of Event Monitor Activations

The number of times an event monitor has been activated.

Element identifier
evmon_activates

Element type
counter

Table 451. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tables	event_table	-
Tablespaces	event_tablespace	-
Bufferpools	event_bufferpool	-
Deadlocks	event_deadlock	-
Deadlocks	event_dlconn	-
Deadlocks with Details	event_detailed_dlconn	-
Statements	evmon_stmt_history	

Usage Use this element to correlate information returned by the above event types. This element is applicable only to write-to-table event monitors. This monitor element is not maintained for event monitors that write to a file or pipe.

Only some types of write-to-table event monitors use the evmon_activates monitor element (the event monitor types that do use this element are listed in the previous table, "Event Monitoring Information"). These event monitors update the evmon_activates column of the SYSCAT.EVENTMONITORS catalog table when activated. This change is logged, so the DATABASE CONFIGURATION will display:

```
All committed transactions have been written to disk = NO
```

If an event monitor is created with the AUTOSTART option, and the first user CONNECTS to the database and immediately DISCONNECTS so that the database is deactivated, a log file will be produced.

evmon_flushes - Number of Event Monitor Flushes

The number of times the FLUSH EVENT MONITOR SQL statement has been issued.

Element identifier
evmon_flushes

Element type
information

Table 452. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tables	event_table	-
Tablespaces	event_tablespace	-
Bufferpools	event_bufferpool	-

Usage This identifier increments with each successive FLUSH EVENT MONITOR

SQL request processed by the database manager after an application has connected to the database. This element helps to uniquely identify database, table, table space and buffer pool data.

executable_id - Executable ID monitor element

An opaque binary token generated on the data server that uniquely identifies the SQL statement section that was executed. For non-SQL activities, a 0-length string value is returned.

Table 453. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	Always collected

Table 454. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activity	event_activitystmt	-
Package cache	-	COLLECT BASE DATA

Usage

Use this monitor element as input to different monitoring interfaces to obtain data about the section. The MON_GET_PKG_CACHE_STMT table function, which is used to get SQL statement activity metrics in the package cache, takes the executable ID as input.

execution_id - User Login ID

The ID that the user specified when logging in to the operating system. This ID is distinct from auth_id, which the user specifies when connecting to the database.

Table 455. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
Application	appl	Basic
DCS Application	dcs_appl_info	Basic

Table 456. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connections	event_connheader	-

Usage You can use this element to determine the operating system userid of the individual running the application that you are monitoring.

failed_sql_stmts - Failed Statement Operations

The number of SQL statements that were attempted, but failed.

Table 457. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Database	dbase_remote	Basic
Application	appl	Basic
Application	appl_remote	Basic
DCS Database	dcs_dbase	Basic
DCS Application	dcs_appl	Basic

For snapshot monitoring, this counter can be reset.

Table 458. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage You can use this element to calculate the total number of successful SQL statements at the database or application level:

```
  dynamic_sql_stmts
+ static_sql_stmts
- failed_sql_stmts
= throughput during monitoring period
```

This count includes all SQL statements that received a negative SQLCODE.

This element may also help you in determining reasons for poor performance, since failed statements mean time wasted by the database manager and as a result, lower throughput for the database.

fcm_message_recv_volume - FCM message received volume monitor element

The amount of data received for internal requests (such as RPCs) distributed by the FCM communications layer. This value is reported in bytes.

Table 459. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 459. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 460. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element to determine how much of the data volume being sent through the FCM subsystem is for request or reply message traffic as opposed to actual table data.

fcm_message_recv_wait_time - FCM message received wait time monitor element

The time spent by an agent waiting for an FCM reply message containing the results of a previously sent FCM request message. This value reflects both the time required to send the response between partitions using FCM and the time required for the subagent to process the request message. The value is given in milliseconds.

Table 461. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 462. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

This element can be used to determine how much time was spent on a given partition waiting for requests to be processed on other partitions in a multi-partition instance.

fcm_message_recvs_total - Total FCM message receives monitor element

The total number of buffers received as part of an FCM reply message containing the results of a previously sent FCM request message.

Table 463. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 464. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

This element can be used to determine both the average volume per FCM message received, as well as the average time spent waiting for a single FCM message to be received.

fcm_message_send_volume - FCM message send volume monitor element

Amount of data volume sent via internal FCM requests. This value is reported in bytes.

Table 465. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 466. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

Table 466. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.		

Usage

Use this element to determine how much of the data volume being sent through the FCM subsystem is used for sending request and reply message traffic, as opposed to sending actual table data.

fcm_message_send_wait_time - FCM message send wait time monitor element

The time spent blocking on an FCM message send. The value is given in milliseconds. This monitor element reflects the time spent blocking for FCM buffers to be flushed from an FCM channel when distributing internal requests on the database system.

Table 467. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 468. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE

Table 468. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element to determine how much time agents are spending waiting to send an FCM request message through the FCM subsystem. Depending on how busy the FCM daemons are, an agent may need to wait when attempting to send messages.

fcm_message_sends_total - Total FCM message sends monitor element

The total number of buffers distributed as part of internal requests using the FCM communications mechanism.

Table 469. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 470. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element to determine both the average amount of data sent per FCM request message, as well as the average amount of time waited per FCM message.

fcm_recv_volume - FCM received volume monitor element

The total amount of data received via the FCM communications layer. This value is reported in bytes.

Table 471. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE

Table 471. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 472. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Indicates the total volume of data received on this partition using FCM, including both message traffic and table queue data.

fcm_recv_wait_time - FCM received wait time monitor element

The total time spent waiting to receive data through FCM. The value is given in milliseconds.

Table 473. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 473. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 474. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element to determine the total time spent waiting to receive data through FCM on this database partition. This includes both data from replies to request messages as well as table queue data.

fcm_recvs_total - FCM receives total monitor element

Total number of buffers received for internal requests using the FCM communications mechanism. The fcm_recvs_total monitor element value is the sum of the values for the fcm_message_recvs_total and fcm_tq_recvs_total monitor elements.

Table 475. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 476. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE

Table 476. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element together with the **fcm_recv_wait_time** monitor element to determine the average wait time per FCM receive operation as well as the average volume returned from an FCM receive operation.

fcm_send_volume - FCM send volume monitor element

The total amount of data distributed by the FCM communications layer. This value is reported in bytes.

Table 477. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 477. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 478. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE

Usage

Use this monitor element to determine the total volume of data sent using FCM, including both message traffic and table queue data.

fcm_send_wait_time - FCM send wait time monitor element

The time spent blocking on an FCM send operation. This includes time spent waiting for buffers for internal requests to be flushed and time spent waiting for window count acknowledgements when sending data over table queues. The value is given in milliseconds.

Table 479. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 479. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 480. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element to determine the total time spent waiting to send data through FCM. This includes both request messages and table queue data.

fcm_sends_total - FCM sends total monitor element

The total number of buffers sent using the internal FCM communications layer.

Table 481. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 482. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE

Table 482. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element to determine the average wait time per FCM receive operation as well as the average volume returned from an FCM receive operation.

fcm_tq_recv_volume - FCM table queue received volume monitor element

The amount of data received on table queues by the FCM communications layer. This value is reported in bytes.

Table 483. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 484. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE

Table 484. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this monitor element to determine the total volume of data received through table queues.

fcm_tq_recv_wait_time - FCM table queue received wait time monitor element

The time spent waiting to receive the next buffer from a table queue. The value is given in milliseconds.

Table 485. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 486. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element to determine how much time agents are spending waiting to receive data on table queues.

fcm_tq_recvs_total - FCM table queue receives total monitor element

The total number of buffers received from table queues using the internal FCM communications mechanism.

Table 487. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 488. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element in conjunction with **fcm_tq_recv_volume** and **fcm_tq_recv_wait_time** to determine the average wait time and volume per table queue buffer received.

fcm_tq_send_volume - FCM table queue send volume monitor element

The amount of data sent over table queues by the FCM communications layer. This value is reported in bytes.

Table 489. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

Table 489. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 490. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this monitor element to determine the total volume of data sent over FCM through table queue buffer sends.

fcm_tq_send_wait_time - FCM table queue send wait time monitor element

The time spent waiting to send the next buffer through a table queue. This reflects the time spent waiting for window count acknowledgements from the receiver end of the table queue. The value is given in milliseconds.

Table 491. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 491. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 492. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this monitor element to determine how much time is being spent waiting to send a data buffer over FCM through a table queue.

fcm_tq_sends_total - FCM table queue send total monitor element

The total number of buffers containing table queue data sent using the internal FCM communications mechanism.

Table 493. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 493. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 494. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element in conjunction with **fcm_tq_send_volume** and **fcm_tq_send_wait_time** monitor elements to determine the average data volume and time waited for buffer sent using a table queue.

fetch_count - Number of Successful Fetches

The number of successful physical fetches or the number of attempted physical fetches, depending on the snapshot monitoring level.

- For the stmt and dynsql snapshot monitoring levels and the statement event type: the number of successful fetches performed on a specific cursor.
- For the dcs_stmt snapshot monitoring level: The number of attempted physical fetches during a statement's execution (regardless of how many rows were

fetched by the application). In this situation, **fetch_count** represents the number of times the server needed to send a reply data back to the gateway while processing a statement.

Table 495. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement
DCS Statement	dcs_stmt	Statement
Dynamic SQL	dynsql	Statement

For Dynamic SQL snapshot monitoring, this counter can be reset.

Table 496. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_stmt	-

Usage

You can use this element to gain insight into the current level of activity within the database manager.

For performance reasons, a statement event monitor does not generate a statement event record for every FETCH statement. A record event is only generated when a FETCH returns a non-zero SQLCODE.

files_closed - Database files closed monitor element

The total number of database files closed.

Table 497. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 498. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 499. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-

Usage

The database manager opens files for reading and writing into and out of the buffer pool. The maximum number of database files open by an application at any time is controlled by the **maxfilop** configuration parameter. If the maximum is reached, one file will be closed before the new file is opened. Note that the actual number of files opened may not equal the number of files closed.

You can use this element to help you determine the best value for the **maxfilop** configuration parameter.

first_active_log - First Active Log File Number

The file number of the first active log file.

Element identifier
first_active_log

Element type
information

Table 500. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	detail_log	Basic

Table 501. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage Use this element in conjunction with the *last_active_log* and *current_active_log* elements to determine the range of active log files. Knowing the range of active log files helps you determine the disk space required for log files.

You can also use this element to determine which log files have data to help you identify log files needed for split mirror support.

first_overflow_time - Time of First Event Overflow

The date and time of the first overflow recorded by this overflow record.

Table 502. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Overflow Record	event_overflow	-

Usage Use this element with *last_over_flow time* to calculate the elapsed time for which the overflow record was generated.

fs_caching - File system caching monitor element

Indicates whether a particular table space uses file system caching. If **fs_caching** is 0, file system caching is enabled. If **fs_caching** is 1, file system caching is disabled.

Table 503. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 504. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table space	tablespace	Basic

Table 505. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Tablespaces	event_tablespace	-

fs_id - Unique file system identification number monitor element

This element shows the unique identification number provided by the operating system for a file system pointed to by a storage path or container.

Table 506. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_CONTAINER table function - Get table space container metrics	Always collected

Table 507. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	db_sto_path_info	Buffer Pool

Usage

Use this element together with the following elements to gather data on space utilization for the database:

- **db_storage_path**
- **sto_path_free_sz**
- **fs_used_size**
- **fs_total_size**

fs_total_size - Total size of a file system monitor element

This element shows the capacity (in bytes) of a file system pointed to by a storage path or container.

Table 508. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_CONTAINER table function - Get table space container metrics	Always collected

Table 509. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	db_sto_path_info	Buffer Pool

Usage

You can use this element together with the following elements to gather data on space utilization for the database:

- **db_storage_path**
- **sto_path_free_sz**
- **fs_used_size**
- **fs_id**

fs_used_size - Amount of space used on a file system monitor element

This element shows the amount of space (in bytes) already used on a file system pointed to by a storage path or container.

Table 510. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_CONTAINER table function - Get table space container metrics	Always collected

Table 511. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	db_sto_path_info	Buffer Pool

Usage

You can use this element together with the following elements to gather data on space utilization for the database:

- **db_storage_path**
- **sto_path_free_sz**
- **fs_total_size**
- **fs_id**

gw_comm_error_time - Communication Error Time

The date and time when the most recent communication error (SQL30081) occurred while a DCS application was attempting to connect to a host database, or while it was processing an SQL statement.

Table 512. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Timestamp

Usage

Use this element for problem determination, in conjunction with Communication Error and the communication error logged in `administration notification log`.

gw_comm_errors - Communication Errors

The number of times that a communication error (SQL30081) occurred while a DCS application was attempting to connect to a host database, or while it was processing an SQL statement.

Table 513. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Basic

For snapshot monitoring, this counter can be reset.

Usage By monitoring the number of communication errors over time, you can assess whether your DB2 Connect gateway has connectivity problems with a particular host database. You can establish what you consider to be a normal error threshold, so that any time the number of errors exceeds this threshold an investigation of the communication errors should be made.

Use this element for problem determination, in conjunction with the communication error logged in `administration notification log`.

gw_con_time - DB2 Connect Gateway First Connect Initiated

The date and time when the first connection to the host database was initiated from the DB2 Connect gateway.

Table 514. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Timestamp
DCS Application	dcs_appl	Timestamp

Usage Use this element for problem determination on DCS applications.

gw_connections_top - Maximum Number of Concurrent Connections to Host Database

The maximum number of concurrent connections to a host database that have been handled by the DB2 Connect gateway since the first database connection.

Element identifier
gw_connections_top

Element type
watermark

Table 515. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Basic

Usage This element will help you understand the level of activity at the DB2 Connect gateway and the associated use of system resources.

gw_cons_wait_client - Number of Connections Waiting for the Client to Send Request

The current number of connections to host databases being handled by the DB2 Connect gateway that are waiting for the client to send a request.

Element identifier
gw_cons_wait_client

Element type
gauge

Table 516. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic
DCS Database	dcs_dbase	Basic

Usage This value can change frequently. It should be sampled at regular intervals over an extended period in order to obtain a realistic view of gateway usage.

gw_cons_wait_host - Number of Connections Waiting for the Host to Reply

The current number of connections to host databases being handled by the DB2 Connect gateway that are waiting for a reply from the host.

Element identifier
gw_cons_wait_host

Element type
gauge

Table 517. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic
DCS Database	dcs_dbase	Basic

Usage This value can change frequently. It should be sampled at regular intervals over an extended period in order to obtain a realistic view of gateway usage.

gw_cur_cons - Current Number of Connections for DB2 Connect

The current number of connections to host databases being handled by the DB2 Connect gateway.

Table 518. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic
DCS Database	dcs_dbase	Basic

Usage This element will help you understand the level of activity at the DB2 Connect gateway and the associated use of system resources.

gw_db_alias - Database Alias at the Gateway

The alias used at the DB2 Connect gateway to connect to the host database.

Element identifier
gw_db_alias

Element type
information

Table 519. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcs_appl_info	Basic

Usage Use this element for problem determination on DCS applications.

gw_exec_time - Elapsed Time Spent on DB2 Connect Gateway Processing

The time in seconds and microseconds at the DB2 Connect gateway to process an application request (since the connection was established), or to process a single statement.

Table 520. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcs_appl	Statement, Timestamp
DCS Statement	dcs_stmt	Statement, Timestamp

For snapshot monitoring, this counter can be reset.

Usage Use this element to determine what portion of the overall processing time is due to DB2 Connect gateway processing.

gw_total_cons - Total Number of Attempted Connections for DB2 Connect

The total number of connections attempted from the DB2 Connect gateway since the last `db2start` command or the last reset.

Table 521. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic
DCS Database	dcs_dbase	Basic

For snapshot monitoring, this counter can be reset.

Usage This element will help you understand the level of activity at the DB2 Connect gateway and the associated use of system resources.

hadr_connect_status - HADR Connection Status monitor element

The current high availability disaster recovery (HADR) connection status of the database.

Table 522. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

Usage

Use this element to determine the HADR connection status of a database.

The data type of this element is integer.

If the database is in HADR primary or standby role, the value for this element is one of the following constants:

SQLM_HADR_CONN_CONNECTED
The database is connected to its partner node.

SQLM_HADR_CONN_DISCONNECTED
The database is not connected to its partner node.

SQLM_HADR_CONN_CONGESTED
The database is connected to its partner node, but the connection is congested. A connection is congested when the TCP/IP socket connection between the primary-standby pair is still alive, but one end cannot send to the other end. For example, the receiving end is not receiving from the socket connection, resulting in a full TCP/IP send space. The reasons for network connection being congested include the following:

- The network is being shared by too many resources or the network is not fast enough for the transaction volume of the primary HADR node.
- The server on which the standby HADR node resides is not powerful enough to retrieve information from the communication subsystem at the necessary rate.

This element should be ignored if the database's HADR role is standard. Use the **hadr_role** monitor element to determine the HADR role of the database.

hadr_connect_time - HADR Connection Time monitor element

Shows one of the following: high availability disaster recovery (HADR) connection time, HADR congestion time, or HADR disconnection time.

Table 523. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

Usage

Use this element to determine when the current HADR connection status began.

If the database is in HADR primary or standby role, the meaning of this element depends on the value of the **hadr_connect_status** element:

- If the value of the **hadr_connect_status** element is SQLM_HADR_CONN_CONNECTED, then this element shows connection time.
- If the value of the **hadr_connect_status** element is SQLM_HADR_CONN_CONGESTED, then this element shows the time when congestion began.
- If the value of the **hadr_connect_status** element is SQLM_HADR_CONN_DISCONNECTED, then this element shows disconnection time.

If there has been no connection since the HADR engine dispatchable unit (EDU) was started, connection status is reported as Disconnected and HADR EDU startup time is used for the disconnection time. Since HADR connect and disconnect events are relatively infrequent, the time is collected and reported even if the DFT_MON_TIMESTAMP switch is off.

This element should be ignored if the database's HADR role is standard. Use the **hadr_role** monitor element to determine the HADR role of the database.

hadr_heartbeat - HADR Heartbeat monitor element

Number of consecutively missed heartbeats on the high availability disaster recovery (HADR) connection. This number is reset to zero when the database receives a heartbeat again. If the database is in the HADR primary or standby role, this element indicates the health of the HADR connection.

Table 524. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

For snapshot monitoring, this counter cannot be reset.

Usage note:

Use this element to determine the health of the HADR connection.

A heartbeat is a message sent from the other HADR database at regular intervals. If the value for this element is zero, no heartbeats have been missed and the connection is healthy. The higher the value, the worse the condition of the connection.

In disconnected mode, heartbeat missed is always shown as 0, because it is not applicable.

The heartbeat interval is derived from configuration parameters such as **hadr_timeout** and **hadr_peer_window**, with a maximal setting of 30 seconds.

The data type of this element is integer.

Ignore this element if the HADR role of the database is standard. Use the **hadr_role** monitor element to determine the HADR role of the database.

hadr_local_host - HADR Local Host monitor element

The local high availability disaster recovery (HADR) host name. The value is displayed as a host name string or an IP address string such as "1.2.3.4".

Table 525. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

Usage

Use this element to determine the effective HADR local host name. HADR database configuration parameters are static. Changes to a parameter are not effective until the database is stopped and restarted. This monitor element reports the value that the HADR system is actually using rather than the value in the database configuration file.

This element should be ignored if the database's HADR role is standard. Use the **hadr_role** monitor element to determine the HADR role of the database.

Note: Any name used must resolve to one IP address. A name that resolves to more than one address will cause an error when trying to start HADR.

hadr_local_service - HADR Local Service monitor element

The local HADR TCP service. This value is displayed as a service name string or a port number string.

Table 526. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

Usage

Use this element to determine the effective HADR local service name. HADR database configuration parameters are static. Changes to a parameter are not

effective until the database is stopped and restarted. This monitor element reports the value the HADR system is actually using rather than the value in the database configuration file.

This element should be ignored if the database's HADR role is standard. Use the **hadr_role** monitor element to determine the HADR role of the database.

hadr_log_gap - HADR Log Gap

This element shows the running average of the gap between the primary Log sequence number (LSN) and the standby log LSN. The gap is measured in number of bytes.

Table 527. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

Usage

Use this element to determine the gap between the primary and standby HADR database logs.

When a log file is truncated, the LSN in the next log file starts as if the last file were not truncated. This LSN hole does not contain any log data. Such holes can cause the log gap not to reflect the actual log difference between the primary and the standby HADR database logs.

This element should be ignored if the database's HADR role is standard. Use the **hadr_role** monitor element to determine the HADR role of the database.

hadr_peer_window - HADR peer window monitor element

The value of the HADR_PEER_WINDOW database configuration parameter.

Table 528. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

Usage

Use this element to determine the value of the HADR_PEER_WINDOW database configuration parameter.

hadr_peer_window_end - HADR peer window end monitor element

The point in time until which a high availability disaster recovery (HADR) primary database promises to stay in peer or disconnected peer state, as long as the primary database is active.

Table 529. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

Usage

Use this element to determine the point in time until which the primary promises to stay in peer or disconnected peer state.

The value reported by the primary database might be different from the value reported by the standby database. This occurs because the primary database updates the value when it sends a heartbeat message, but the new value is shown on the standby database only after the message is received and processed on the standby database.

If a database moves out of peer or disconnected peer state, the value of this monitor element is not reset. The last known value is kept and returned. If a database never reached peer state, a value of zero will be returned.

The peer window end time is set by the primary database and then sent to the standby database. For this reason, the value of the peer window end is based on the clock of the primary database. When you compare the peer window end time with the primary database down time, you might need to add an offset to convert the timestamp to the primary database clock, if the two clocks are not well synchronized.

hadr_primary_log_file - HADR Primary Log File monitor element

The name of the current log file on the primary HADR database.

Table 530. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

Usage

Use this element to determine the current log file on the primary HADR database.

This element should be ignored if the database's HADR role is standard. Use the **hadr_role** monitor element to determine the HADR role of the database.

hadr_primary_log_lsn - HADR Primary Log LSN monitor element

The current log position of the primary HADR database. Log sequence number (LSN) is a byte offset in the database's log stream.

Table 531. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

Usage

Use this element to determine the current log position on the primary HADR database.

This element should be ignored if the database's HADR role is standard. Use the **hadr_role** monitor element to determine the HADR role of the database.

hadr_primary_log_page - HADR Primary Log Page monitor element

The page number in the current log file indicating the current log position on the primary HADR database. The page number is relative to the log file. For example, page zero is the beginning of the file.

Table 532. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

Usage

Use this element to determine the current log page on the primary HADR database.

This element should be ignored if the database's HADR role is standard. Use the **hadr_role** monitor element to determine the HADR role of the database.

hadr_remote_host - HADR Remote Host monitor element

The remote high availability disaster recovery (HADR) host name. The value is displayed as a host name string or an IP address string such as "1.2.3.4".

Table 533. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

Usage

Use this element to determine the effective HADR remote host name. HADR database configuration parameters are static. Changes to a parameter are not effective until the database is stopped and restarted. This monitor element reports the value the HADR system is actually using rather than the value in the database configuration file.

This element should be ignored if the database's HADR role is standard. Use the *hadr_role* monitor element to determine the HADR role of the database.

Note: Any name used must resolve to one IP address. A name that resolves to more than one address will cause an error when trying to start HADR.

hadr_remote_instance - HADR Remote Instance monitor element

The remote HADR instance name.

Table 534. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

Usage

Use this element to determine the effective HADR remote instance name. HADR database configuration parameters are static. Changes to a parameter are not effective until the database is stopped and restarted. This monitor element reports the value the HADR system is actually using rather than the value in the database configuration file.

This element should be ignored if the database's HADR role is standard. Use the **hadr_role** monitor element to determine the HADR role of the database.

hadr_remote_service - HADR Remote Service monitor element

The remote HADR TCP service. This value is displayed as a service name string or a port number string.

Table 535. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

Usage

Use this element to determine the effective HADR remote service name. HADR database configuration parameters are static. Changes to a parameter are not effective until the database is stopped and restarted. This monitor element reports the value the HADR system is actually using rather than the value in the database configuration file.

This element should be ignored if the database's HADR role is standard. Use the **hadr_role** monitor element to determine the HADR role of the database.

hadr_role - HADR Role

The current high availability disaster recovery (HADR) role of the database.

Table 536. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

Usage

Use this element to determine the HADR role of a database.

The data type of this element is integer.

The value for this element is one of the following constants:

SQLM_HADR_ROLE_STANDARD
The database is not an HADR database.

SQLM_HADR_ROLE_PRIMARY
The database is the primary HADR database.

SQLM_HADR_ROLE_STANDBY
The database is the standby HADR database.

hadr_standby_log_file - HADR Standby Log File monitor element

The name of the current log file on the standby HADR database.

Table 537. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

Usage

Use this element to determine the current log file on the standby HADR database.

This element should be ignored if the database's HADR role is standard. Use the **hadr_role** monitor element to determine the HADR role of the database.

hadr_standby_log_lsn - HADR Standby Log LSN monitor element

The current log position of the standby HADR database. Log sequence number (LSN) is a byte offset in the database's log stream.

Table 538. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

Usage

Use this element to determine the current log position on the standby HADR database.

This element should be ignored if the database's HADR role is standard. Use the **hadr_role** monitor element to determine the HADR role of the database.

hadr_standby_log_page - HADR Standby Log Page monitor element

The page number in the current log file indicating the current log position on the standby HADR database. The page number is relative to the log file. For example, page zero is the beginning of the file.

Table 539. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

Usage

Use this element to determine the current log page on the standby HADR database.

This element should be ignored if the database's HADR role is standard. Use the **hadr_role** monitor element to determine the HADR role of the database.

hadr_state - HADR State monitor element

The current high availability disaster recovery (HADR) state of the database.

Table 540. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

Usage

Use this element to determine the HADR state of a database.

The data type of this element is integer. If the database is in HADR primary or standby role, the value for this element is one of the following constants:

SQLM_HADR_STATE_DISCONNECTED
The database is not connected to its partner database.

SQLM_HADR_STATE_LOC_CATCHUP
The database is doing local catchup.

SQLM_HADR_STATE_REM_CATCH_PEND
The database is waiting to connect to its partner to do remote catchup.

SQLM_HADR_STATE_REM_CATCHUP
The database is doing remote catchup.

SQLM_HADR_STATE_PEER
The primary and standby databases are connected and are in peer state.

SQLM_HADR_STATE_DISCONN_PEER
The primary and standby databases are in disconnected peer state.

This element should be ignored if the database's HADR role is standard. Use the **hadr_role** monitor element to determine the HADR role of the database.

hadr_syncmode - HADR Synchronization Mode monitor element

The current high availability disaster recovery (HADR) synchronization mode of the database.

Table 541. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

Usage

Use this element to determine the HADR synchronization mode of a database.

The data type of this element is integer.

HADR database configuration parameters are static. Changes to a parameter are not effective until the database is stopped and restarted. This monitor element reports the value the HADR system is actually using rather than the value in the database configuration file.

If the database is in HADR primary or standby role, the value for this element is one of the following constants:

SQLM_HADR_SYNCMODE_SYNC
SYNC mode.

SQLM_HADR_SYNCMODE_NEARSYNC
NEARSYNC mode.

SQLM_HADR_SYNCMODE_ASYNC
ASYNC mode.

SQLM_HADR_SYNCMODE_SUPERASYNC
SUPERASYNC mode.

This element should be ignored if the database's HADR role is standard. Use the **hadr_role** monitor element to determine the HADR role of the database.

hadr_timeout - HADR Timeout monitor element

The number of seconds without any communication from its partner after which an HADR database server will consider that the connection between them has failed.

Table 542. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

Usage

Use this element to determine the effective HADR timeout value. HADR database configuration parameters are static. Changes to a parameter are not effective until the database is stopped and restarted. This monitor element reports the value the HADR system is actually using rather than the value in the database configuration file.

This element should be ignored if the database's HADR role is standard. Use the **hadr_role** monitor element to determine the HADR role of the database.

hash_join_overflows - Hash Join Overflows

The number of times that hash join data exceeded the available sort heap space.

Element identifier
hash_join_overflows

Element type
counter

Table 543. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 544. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage At the database level, if the value of hash_join_small_overflows is greater than 10% of this hash_join_overflows, then you should consider increasing the sort heap size. Values at the application level can be used to evaluate hash join performance for individual applications.

hash_join_small_overflows - Hash Join Small Overflows

The number of times that hash join data exceeded the available sort heap space by less than 10%.

Element identifier
hash_join_small_overflows

Element type
counter

Table 545. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 546. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage If this value and hash_join_overflows are high, then you should consider increasing the sort heap threshold. If this value is greater than 10% of hash_join_overflows, then you should consider increasing the sort heap size.

histogram_type - Histogram type monitor element

The type of the histogram, in string format.

There are six histogram types.

CoordActQueueTime
A histogram of the time non-nested activities spend queued (for example, in a threshold queue), measured on the coordinator partition.

CoordActExecTime
A histogram of the time non-nested activities spend executing at the coordinator partition. Execution time does not include time spent initializing or queued. For cursors, execution time includes only the time spent on open, fetch and close requests. When an activity is remapped

between service subclasses, the execution time histogram is updated only for the service subclass in which the activity completes execution.

CoordActLifetime
A histogram of the elapsed time from when a non-nested activity is identified by the database manager until the activity completes execution, as measured on the coordinator partition. When you remap activities between service subclasses, the lifetime histogram is updated only for the service subclass in which the activity completes execution.

CoordActInterArrivalTime
A histogram of the time interval between the arrival of non-nested coordinator activities. The inter-arrival time mean is calculated for service subclasses through which activities enter the system. When you remap activities between service subclasses, the inter-arrival time histogram of the service subclass you remap an activity to is unaffected.

CoordActEstCost
A histogram of the estimated cost of non-nested DML activities. The estimated cost of an activity is counted only towards the service subclass in which the activity enters the system.

ReqExecTime
A histogram of request execution times, which includes requests on the coordinator partition, and any subrequests on both coordinator and non-coordinator partitions (like RPC requests or SMP subagent requests). Requests included may or may not be associated with an activity: Both PREPARE and OPEN requests are included in this histogram, for example, but while OPEN requests are always associated with a cursor activity, PREPARE requests are not part of any activity.The execution time histogram of a service subclass involved in remapping counts the portion of the execution time spent by the partial request in the service subclass.

Table 547. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_histogrambin	-

Usage

Use this element to identify the type of histogram. Several histograms can belong to the same statistics record, but only one of each type.

hld_application_handle - Identifier for the application holding the lock monitor element

System-wide unique ID for the application that is holding the lock. If the application holding this lock is unknown or cannot be found, then a value of NULL is returned.

Table 548. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_APPL_LOCKWAIT table function - get information about locks for which an application is waiting	

hld_member - Database member for application holding lock

Database member where the lock is being held by the holding application.

Table 549. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_APPL_LOCKWAIT table function - get information about locks for which an application is waiting	

Usage

If the lock is being held on a remote member, the value of **hld_member** is -2. To determine which member the lock is being held at, use the MON_GET_LOCKS table function and specify the **lock_name** as a search argument.

host_ccsid - Host Coded Character Set ID

This is the coded character set identifier (CCSID) of the host database.

Element identifier
host_ccsid

Element type
information

Table 550. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcs_appl_info	Basic

Usage Use this element for problem determination on DCS applications.

host_db_name - Host Database Name

The real name of the host database for which information is being collected or to which the application is connected. This is the name that was given to the database when it was created.

Table 551. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Basic
DCS Application	dcs_appl_info	Basic

Usage Use this element for problem determination on DCS applications.

host_prdid - Host Product/Version ID

The product and version that is running on the server.

Table 552. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcs_appl_info	Basic

Usage Used to identify the product and code version of the DRDA host database product. It is in the form PPPVVRRM, where:

- PPP identifies the host DRDA product
  - ARI for DB2 Server for VSE & VM
  - DSN for DB2 for z/OS
  - QSQ for DB2 for i
  - SQL for other DB2 products.
- VV identifies a 2-digit version number (with high-order 0 in the case of a 1-digit version)
- RR identifies a 2-digit release number (with high-order 0 in the case of a 1-digit release)
- M identifies a 1-character modification level (0-9 or A-Z)

host_response_time - Host Response Time

At the DCS statement level, this is the elapsed time between the time that the statement was sent from the DB2 Connect gateway to the host for processing and the time when the result was received from the host. At DCS database and DCS application levels, it is the sum of the elapsed times for all the statements that were executed for a particular application or database. At the data transmission level, this is the sum of host response times for all the statements that used this many data transmissions.

Table 553. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement, Timestamp
DCS Statement	dcs_stmt	Statement, Timestamp
Data Transmission	stmt_transmissions	Statement, Timestamp

For snapshot monitoring at the statement level, this counter cannot be reset. This counter can be reset at other levels.

Usage

Use this element with Outbound Number of Bytes Sent and Outbound Number of Bytes Received to calculate the outbound response time (transfer rate):

```
(outbound bytes sent + outbound bytes received) / host response time
```

This element is composed of two subelements that report time spent as seconds and microseconds (one millionth of a second). The names of the subelements can be derived by adding "_s" and "_ms" to the name of this monitor element. To retrieve the total time spent for this monitor element, the values of the two subelements must be added together. For example, if the "_s" subelement value is 3 and the "_ms" subelement value is 20, then the total time spent for the monitor element is 3.00002 seconds.

hostname - Host name monitor element

The host name of the machine on which the database member resides.

Table 554. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_FCM - Get FCM metrics	Always collected

host_name - Host name monitor element

The host name of the machine on which the database member or partition resides.

Alias of **hostname** monitor element.

Table 555. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_NETWORK_RESOURCES table function - Return network adapter information	Always collected
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected
MON_GET_MEMORY_POOL table function - Get memory pool information	Always collected
MON_GET_MEMORY_SET table function - Get memory set information	Always collected
DB_PARTITIONS table function - List information about database partitions	Always collected

ida_recv_volume - Total data volume received monitor element

The total volume of data (in bytes) that the database server received from an in-database analytics process.

Table 556. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
EVMON_FORMAT_UE_TO_XML table function - convert unformatted events to XML	REQUEST METRICS BASE
MON_FORMAT_XML_METRICS_BY_ROW - Get formatted row-based output for all metrics	REQUEST METRICS BASE
MON_FORMAT_XML_TIMES_BY_ROW - Get formatted row-based combined hierarchy wait and processing times	REQUEST METRICS BASE
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - Get formatted row-based output for wait times	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details	ACTIVITY METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE

Table 556. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS - get detailed metrics for package cache entries (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 557. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activity	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activity	event_activitymetrics	ACTIVITY METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Package cache	event_pkgcache_metrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Unit of work	event_uow_metrics	REQUEST METRICS BASE

ida_recv_wait_time - Time spent waiting to receive data monitor element

The total amount of time spent waiting to receive data from an in-database analytics process.

Table 558. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
EVMON_FORMAT_UE_TO_XML table function - convert unformatted events to XML	REQUEST METRICS BASE
MON_FORMAT_XML_METRICS_BY_ROW - Get formatted row-based output for all metrics	REQUEST METRICS BASE
MON_FORMAT_XML_TIMES_BY_ROW - Get formatted row-based combined hierarchy wait and processing times	REQUEST METRICS BASE
MON_FORMAT_XML_WAIT_TIMES_ BY_ROW - Get formatted row-based output for wait times	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details	ACTIVITY METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS - get detailed metrics for package cache entries (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 559. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activity	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activity	event_activitymetrics	ACTIVITY METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Package cache	event_pkgcache_metrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Unit of work	event_uow_metrics	REQUEST METRICS BASE

ida_recvs_total - Number of times data received monitor element

The total number of times data was received from an in-database analytics process.

Table 560. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
EVMON_FORMAT_UE_TO_XML table function - convert unformatted events to XML	REQUEST METRICS BASE
MON_FORMAT_XML_METRICS_BY_ROW - Get formatted row-based output for all metrics	REQUEST METRICS BASE
MON_FORMAT_XML_TIMES_BY_ROW - Get formatted row-based combined hierarchy wait and processing times	REQUEST METRICS BASE
MON_FORMAT_XML_WAIT_TIMES _BY_ROW - Get formatted row-based output for wait times	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details	ACTIVITY METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS - get detailed metrics for package cache entries (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 560. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 561. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activity	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activity	event_activitymetrics	ACTIVITY METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Package cache	event_pkgcache_metrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Unit of work	event_uow_metrics	REQUEST METRICS BASE

ida_send_volume - Total data volume sent monitor element

The total volume of data (in bytes) sent from the database server to an in-database analytics process.

Table 562. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
EVMON_FORMAT_UE_TO_XML table function - convert unformatted events to XML	REQUEST METRICS BASE
MON_FORMAT_XML_METRICS_BY_ROW - Get formatted row-based output for all metrics	REQUEST METRICS BASE
MON_FORMAT_XML_TIMES_BY_ROW - Get formatted row-based combined hierarchy wait and processing times	REQUEST METRICS BASE

Table 562. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_FORMAT_XML_WAIT_TIMES _BY_ROW - Get formatted row-based output for wait times	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details	ACTIVITY METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS - get detailed metrics for package cache entries (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 563. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activity	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activity	event_activitymetrics	ACTIVITY METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Package cache	event_pkgcache_metrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE

Table 563. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Unit of work	event_uow_metrics	REQUEST METRICS BASE

ida_send_wait_time - Time spent waiting to send data monitor element

The total amount of time spent waiting to send data to an in-database analytics process.

Table 564. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
EVMON_FORMAT_UE_TO_XML table function - convert unformatted events to XML	REQUEST METRICS BASE
MON_FORMAT_XML_METRICS_BY_ROW - Get formatted row-based output for all metrics	REQUEST METRICS BASE
MON_FORMAT_XML_TIMES_BY_ROW - Get formatted row-based combined hierarchy wait and processing times	REQUEST METRICS BASE
MON_FORMAT_XML_WAIT_TIMES _BY_ROW - Get formatted row-based output for wait times	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details	ACTIVITY METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS - get detailed metrics for package cache entries (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 564. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 565. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activity	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activity	event_activitymetrics	ACTIVITY METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Package cache	event_pkgcache_metrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Unit of work	event_uow_metrics	REQUEST METRICS BASE

ida_sends_total - Number of times data sent monitor element

The total number of times data was sent to an in-database analytics process.

Table 566. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
EVMON_FORMAT_UE_TO_XML table function - convert unformatted events to XML	REQUEST METRICS BASE
MON_FORMAT_XML_METRICS_BY_ROW - Get formatted row-based output for all metrics	REQUEST METRICS BASE
MON_FORMAT_XML_TIMES_BY_ROW - Get formatted row-based combined hierarchy wait and processing times	REQUEST METRICS BASE
MON_FORMAT_XML_WAIT_TIMES _BY_ROW - Get formatted row-based output for wait times	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details	ACTIVITY METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 566. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS - get detailed metrics for package cache entries (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 567. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activity	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activity	event_activitymetrics	ACTIVITY METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Package cache	event_pkgcache_metrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Unit of work	event_uow_metrics	REQUEST METRICS BASE

idle_agents - Number of Idle Agents

The number of agents in the agent pool that are currently unassigned to an application and are, therefore, "idle".

Table 568. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

Usage You can use this element to help set the *num_poolagents* configuration parameter. Having idle agents available to service requests for agents can improve performance.

iid - Index identifier monitor element

Identifier for the index.

Table 569. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

inbound_bytes_received - Inbound Number of Bytes Received

The number of bytes received by the DB2 Connect gateway from the client, excluding communication protocol overhead (for example, TCP/IP or SNA headers).

Table 570. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcs_appl	Basic
DCS Statement	dcs_stmt	Statement

For snapshot monitoring at the application level, this counter can be reset. This counter cannot be reset at other levels.

Usage Use this element to measure the throughput from the client to the DB2 Connect gateway.

inbound_bytes_sent - Inbound Number of Bytes Sent

The number of bytes sent by the DB2 Connect gateway to the client, excluding communication protocol overhead (for example, TCP/IP or SNA headers).

Table 571. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcs_appl	Basic
DCS Statement	dcs_stmt	Statement

For snapshot monitoring at the application level, this counter can be reset. This counter cannot be reset at other levels.

Usage Use this element to measure the throughput from the DB2 Connect gateway to the client.

inbound_comm_address - Inbound Communication Address

This is the communication address of the client. For example, it could be an SNA net ID and LU partner name, or an IP address and port number for TCP/IP.

Table 572. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
DCS Application	dcs_appl_info	Basic

Usage Use this element for problem determination on DCS applications.

When returned by the SNAPSHOT_APPL table function, the SNAP_GET_APPL table function, or the SNAP_GET_APPL_V95 table function, this monitor element will truncate IPv6 addresses whose values are larger than 32 bits.

include_col_updates - Include column updates monitor element

Number of include column updates.

Table 573. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

index_object_pages - Index Object Pages

The number of disk pages consumed by all indexes defined on a table.

Table 574. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table	Basic

Table 575. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Tables	event_table	-

Usage This element provides a mechanism for viewing the actual amount of space consumed by indexes defined on a particular table. This element can be used in conjunction with a table event monitor to track the rate of index growth over time. This element is not returned for partitioned tables.

index_object_l_pages - Index data logical pages monitor element

The number of logical pages used on disk by all indexes associated with this table. For partitioned tables, the value returned is NULL.

Table 576. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLE table function - Get table metrics	DATA OBJECT METRICS BASE

Usage

- This value might be less than the amount of space physically allocated for the object. This can happen when you use the REUSE STORAGE option of the TRUNCATE statement. This option causes storage allocated for the table to continue to be allocated, although the storage will be considered empty. In addition, the value for this monitor element might be less than the amount of space logically allocated for the object, because the total space logically allocated includes a small amount of additional meta data.

  To retrieve an accurate measure of the logical or physical size of an object, use the ADMIN_GET_TAB_INFO_V97 function. This function provides more accurate information about the size of objects than you can obtain by multiplying the number of pages reported for this monitor element by the page size.

index_only_scans - Index-only scans monitor element

Number of index-only scans. Index-only scans occur when the results of scan was satisfied by access to the index only.

Table 577. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

index_scans - Index scans monitor element

Number of index scans.

Table 578. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

index_tbsp_id - Index table space ID monitor element

An identifier of the table space that holds indexes created on this table.

Table 579. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLE table function - Get table metrics	Always collected

Usage

The value of this element matches a value from column TBSPACEID of view SYSCAT.TABLESPACES.

input_db_alias - Input Database Alias

The alias of the database provided when calling the snapshot function.

Table 580. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl_id_info	Basic
Table Space	tablespace_list	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Table	table_list	Table
Lock	db_lock_list	Basic

Usage This element can be used to identify the specific database to which the monitor data applies. It contains blanks unless you requested monitor information related to a specific database.

The value of this field may be different than the value of the *client_db_alias* monitor element since a database can have many different aliases. Different applications and users can use different aliases to connect to the same database.

insert_sql_stmts - Inserts

This element contains a count of the total number of times the federated server has issued an INSERT statement to this data source on behalf of any application since the start of the federated server instance or the last reset of the database monitor counters, whichever is latest.

Table 581. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

Usage

Use this element to determine the level of database activity directed against this data source by the federated server or an application.

You can also use this element to determine the percentage of write activity against this data source by the federated server or an application, with the following formula:

```
write_activity =
   (INSERT statements + UPDATE statements + DELETE statements ) /
   (SELECT statements + INSERT statements + UPDATE statements +
   DELETE statements)
```

insert_time - Insert Response Time

This element contains the aggregate amount of time, in milliseconds, that it has taken this data source to respond to INSERTs from all applications or a single application running on this federated server instance since the start of the federated server instance or the last reset of the database monitor counters, whichever is latest.

The response time is measured as the difference in time between the time the federated server submits an INSERT statement to the data source, and the time the data source responds to the federated server, indicating that the INSERT has been processed.

Table 582. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Timestamp
Application	appl_remote	Timestamp

For snapshot monitoring, this counter can be reset.

Usage

Use this element to determine the actual amount of time that transpires waiting for INSERTs to this data source to be processed. This information can be useful for capacity planning and tuning.

insert_timestamp - Insert timestamp monitor element

The time when the statement or section was inserted into the cache. For dynamic sql snapshots this represents the time when the statement entered the cache. For MON_GET_PKG_CACHE_STMT, MON_GET_PKG_CACHE_STMT_DETAILS and the package cache event monitor, the value is more granular and represents the time when an individual section for this statement was inserted into the cache.

Table 583. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	Always collected

Table 584. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Dynamic SQL	dynsql	Basic

Table 585. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Package cache	-	COLLECT BASE DATA

Usage

This element specifies the time when the statement was inserted into the cache. It can be used to estimate the lifetime of a statement in the cache.

int_auto_rebinds - Internal Automatic Rebinds

The number of automatic rebinds (or recompiles) that have been attempted.

Table 586. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 587. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage Automatic rebinds are the internal binds the system performs when an package has been invalidated. The rebind is performed the first time that the database manager needs to execute an SQL statement from the package. For example, packages are invalidated when you:

- Drop an object, such as a table, view, or index, on which the plan is dependent
- Add or drop a foreign key
- Revoke object privileges on which the plan is dependent.

You can use this element to determine the level of database activity at the application or database level. Since int_auto_rebinds can have a significant impact on performance, they should be minimized where possible.

You can also use this element to determine the percentage of rebind activity using the following formula:

```
int_auto_rebinds / total number of statements
```

This information can be useful for analyzing application activity and throughput.

int_commits - Internal commits monitor element

The total number of commits initiated internally by the database manager.

Table 588. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 589. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 590. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

Table 590. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

An internal commit may occur during any of the following:

- A reorganization
- An import
- A bind or pre-compile
- An application ends without executing an explicit SQL COMMIT statement (on UNIX).

This value, which does not include explicit SQL COMMIT statements, represents the number of these internal commits since the later of:

- The connection to the database (for database-level information, this is the time of the first connection)
- The last reset of the database monitor counters.

You can use this element to calculate the total number of units of work by calculating the sum of the following:

```
  commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks
```

Note: The units of work calculated will only include those since the later of:

- The connection to the database (for database-level information, this is the time of the first connection)
- The last reset of the database monitor counters.

This calculation can be done at the application or the database level.

int_deadlock_rollbacks - Internal Rollbacks Due To Deadlock

The total number of forced rollbacks initiated by the database manager due to a deadlock. A rollback is performed on the current unit of work in an application selected by the database manager to resolve the deadlock.

Element identifier
int_deadlock_rollbacks

Element type
counter

Table 591. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 592. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

Usage This element shows the number of deadlocks that have been broken and can be used as an indicator of concurrency problems. It is important, since int_deadlock_rollbacks lower the throughput of the database.

This value is included in the value given by int_rollbacks.

int_node_splits - Intermediate node splits monitor element

Number of times an intermediate index node was split during an insert operation.

Table 593. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

int_rollbacks - Internal rollbacks monitor element

The total number of rollbacks initiated internally by the database manager.

Table 594. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 595. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 596. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

An internal rollback occurs when any of the following cannot complete successfully:

- A reorganization
- An import
- A bind or pre-compile
- An application ends as a result of a deadlock situation or lock timeout situation
- An application ends without executing an explicit commit or rollback statement (on Windows).

This value represents the number of these internal rollbacks since the later of:

- The connection to the database (for database-level information, this is the time of the first connection)
- The last reset of the database monitor counters.

While this value does not include explicit SQL ROLLBACK statements, the count from the **int_deadlock_rollbacks** monitor element is included.

You can use this element to calculate the total number of units of work by calculating the sum of the following:

```
  commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks
```

Note: The units of work calculated will include those since the later of:

- The connection to the database (for database-level information, this is the time of the first connection)
- The last reset of the database monitor counters.

This calculation can be done at the application or the database level.

int_rows_deleted - Internal Rows Deleted

This is the number of rows deleted from the database as a result of internal activity.

Table 597. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic
Application	stmt	Basic
Dynamic SQL	dynsql	Statement

For snapshot monitoring, this counter can be reset.

Table 598. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-
Statements	event_stmt	-

Usage This element can help you gain insight into internal activity within the database manager of which you might not be aware. If this activity is high, you may want to evaluate your table design to determine if the referential constraints or triggers that you have defined on your database are necessary.

Internal delete activity can be a result of:

- A cascading delete enforcing an ON CASCADE DELETE referential constraint
- A trigger being fired.

int_rows_inserted - Internal Rows Inserted

The number of rows inserted into the database as a result of internal activity caused by triggers.

Table 599. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic
Application	stmt	Basic
Dynamic SQL	dynsql	Statement

For snapshot monitoring, this counter can be reset.

Table 600. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-
Statements	event_stmt	-

Usage This element can help you gain insight into the internal activity within the database manager. If this activity is high, you may want to evaluate your design to determine if you can alter it to reduce this activity.

int_rows_updated - Internal Rows Updated

This is the number of rows updated from the database as a result of internal activity.

Table 601. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic
Application	stmt	Basic
Dynamic SQL	dynsql	Statement

For snapshot monitoring, this counter can be reset.

Table 602. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-
Statements	event_stmt	-

Usage This element can help you gain insight into internal activity within the database manager of which you might not be aware. If this activity is high, you may want to evaluate your table design to determine if the referential constraints that you have defined on your database are necessary.

Internal update activity can be a result of:

- A *set null* row update enforcing a referential constraint defined with the ON DELETE SET NULL rule
- A trigger being fired.

invocation_id - Invocation ID monitor element

An identifier that distinguishes one invocation of a routine from others at the same nesting level within a unit of work. It is unique within a unit of work for a specific nesting level.

The **invocation_id** monitor element is an alias of the **stmt_invocation_id** monitor element.

Table 603. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 604. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activitystmt	-
Locking	-	-
Deadlocks with Details History Values[1]	event_stmt_history	-
Deadlocks with Details History[1]	event_stmt_history	-
Unit of work	Reported in the package list.	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

You can use this element to uniquely identify the invocation in which a particular SQL statement has been executed. You can also use this element in conjunction with other statement history entries to see the sequence of SQL statements that caused the deadlock.

ipc_recv_volume - Interprocess communication received volume monitor element

The amount of data received by data server from clients over IPC. This value is reported in bytes.

Table 605. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 605. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 606. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

ipc_recv_wait_time - Interprocess communication received wait time monitor element

The time spent by an agent receiving an incoming client request using the IPC communications protocol. The value is reported in milliseconds.

Table 607. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 607. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 608. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

ipc_recvs_total - Interprocess communication receives total monitor element

The number of times data was received by the database server from the client application using IPC.

Table 609. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 609. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 610. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

ipc_send_volume - Interprocess communication send volume monitor element

The amount of data sent by data server to clients over the IPC protocol. This value is reported in bytes.

Table 611. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 611. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 612. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

ipc_send_wait_time - Interprocess communication send wait time monitor element

The time spent blocking on an IPC send to the client. The value is given in milliseconds.

Table 613. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 613. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 614. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

ipc_sends_total - Interprocess communication send total monitor element

The number of times data was sent by the database server to client applications using IPC.

Table 615. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 615. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 616. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

is_system_appl - Is System Application monitor element

Indicates whether the application is a system application.

Table 617. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic

Usage

The **is_system_appl** monitor element indicates whether an application is an internal system application. The possible values are

0 user application

1 system application

key_updates - Key updates monitor element

Number of key updates.

Table 618. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

last_active_log - Last Active Log File Number

The file number of the last active log file.

Element identifier
last_active_log

Element type
information

Table 619. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	detail_log	Basic

Table 620. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage Use this element in conjunction with the *first_active_log* and *current_active_log* elements to determine the range of active log files. Knowing the range of active log files helps you determine the disk space required for log files.

You can also use this element to determine which log files have data to help you identify log files needed for split mirror support.

last_backup - Last Backup Timestamp

The date and time that the latest database backup was completed.

Element identifier
last_backup

Element type
timestamp

Table 621. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Timestamp

Usage You may use this element to help you identify a database that has not been backed up recently, or to identify which database backup file is the most recent. If the database has never been backed up, this timestamp is initialized to zero.

last_executable_id - Last executable identifier monitor element

The executable id for the statement most recently completed by the application.

Table 622. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected

last_extent - Last extent moved monitor element

The numeric identifier of the last extent moved by the table space rebalancer process.

Table 623. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_EXTENT_MOVEMENT_STATUS - Get extent movement progress status metrics	Always collected

last_metrics_update - Metrics last update timestamp monitor element

Timestamp reflecting the last time metrics were updated for this cache entry.

Table 624. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 625. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Package cache	-	COLLECT BASE DATA

last_overflow_time - Time of Last Event Overflow

The date and time of the last overflow recorded this overflow record.

Table 626. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Overflow Record	event_overflow	-

Usage Use this element with *first_overflow_time* to calculate the elapsed time for which the overflow record was generated.

last_reference_time - Last reference time monitor element

The last time the activity was accessed by a request.

Table 627. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

last_request_type - Last request type monitor element

The type of the last request completed by the application.

Table 628. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected

Usage

This monitor element is only reported on the coordinator member for the application.

The following values are possible.

- CLOSE
- COMMIT
- COMPILE
- DESCRIBE
- EXCSQLSET
- EXECIMMD
- EXECUTE
- FETCH
- INTERNAL *number*, where *number* is the value of the internal constant
- OPEN
- PREPARE
- REBIND
- REDISTRIBUTE
- REORG
- ROLLBACK

- RUNSTATS

last_reset - Last Reset Timestamp

Indicates the date and time that the monitor counters were reset for the application issuing the GET SNAPSHOT.

Element identifier
last_reset

Element type
timestamp

Table 629. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Timestamp
Database	dbase	Timestamp
Application	appl	Timestamp
Table Space	tablespace_list	Buffer Pool, Timestamp
Table	table_list	Timestamp
DCS Database	dcs_dbase	Timestamp
DCS Application	dcs_appl	Timestamp

Usage You can use this element to help you determine the scope of information returned by the database system monitor.

If the counters have never been reset, this element will be zero.

The database manager counters will only be reset if you reset all active databases.

last_wlm_reset - Time of last reset monitor element

This element, in the form of a local timestamp, shows the time at which the last statistics event record of this type was created.

Table 630. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wlstats	-
Statistics	event_wcstats	-
Statistics	event_qstats	-

Usage

Use the **wlm_last_reset** and **statistics_timestamp** monitor elements to determine a period of time over which the statistics in an event monitor statistics record were collected. The collection interval begins at the **wlm_last_reset** time and ends at **statistics_timestamp**.

lob_object_pages - LOB Object Pages

The number of disk pages consumed by LOB data.

Table 631. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table	Basic

Table 632. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Tables	event_table	-

Usage This element provides a mechanism for viewing the actual amount of space consumed by LOB data in a particular table. This element can be used in conjunction with a table event monitor to track the rate of LOB data growth over time.

lob_object_l_pages - LOB data logical pages monitor element

The number of logical pages used on disk by LOBs associated with this table.

Table 633. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLE table function - Get table metrics	DATA OBJECT METRICS BASE

Usage

- This value might be less than the amount of space physically allocated for the object. This can happen when you use the REUSE STORAGE option of the TRUNCATE statement. This option causes storage allocated for the table to continue to be allocated, although the storage will be considered empty. In addition, the value for this monitor element might be less than the amount of space logically allocated for the object, because the total space logically allocated includes a small amount of additional meta data.

  To retrieve an accurate measure of the logical or physical size of an object, use the ADMIN_GET_TAB_INFO_V97 function. This function provides more accurate information about the size of objects than you can obtain by multiplying the number of pages reported for this monitor element by the page size.

local_cons - Local Connections

The number of local applications that are currently connected to a database within the database manager instance being monitored.

Table 634. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

Usage

This number can help you determine the level of concurrent processing occurring in the database manager. This value will change frequently, so you may need to sample it at specific intervals over an extended period of time to get a realistic view of system usage.

This number only includes applications that were initiated from the same instance as the database manager. The applications are connected, but may or may not be executing a unit of work in the database.

When used in conjunction with the rem_cons_in monitor element, this element can help you adjust the setting of the **max_connections** configuration parameter.

local_cons_in_exec - Local Connections Executing in the Database Manager

The number of local applications that are currently connected to a database within the database manager instance being monitored and are currently processing a unit of work.

Table 635. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

Usage

This number can help you determine the level of concurrent processing occurring in the database manager. This value will change frequently, so you may need to sample it at specific intervals over an extended period of time to get a realistic view of system usage. This number only includes applications that were initiated from the same instance as the database manager.

When used in conjunction with the **rem_cons_in_exec** monitor element, this element can help you adjust the setting of the **max_coordagents** configuration parameter.

The following recommendations apply to non-concentrator configurations only. When concentrator is enabled, DB2 is multiplexing a larger number of client connections onto a smaller pool of coordinator agents. In this case, it is usually acceptable to have the sum of **rem_cons_in_exec** and **local_cons_in_exec** approach the **max_coordagents** value.

- If **max_coordagents** is set to AUTOMATIC, do not make any adjustments.
- If **max_coordagents** is not set to AUTOMATIC and if the sum of **rem_cons_in_exec** and **local_cons_in_exec** is close to **max_coordagents**, increase the value of **max_coordagents**.

local_start_time - Local start time monitor element

The time that this activity began doing work on the partition. It is in local time. This field can be an empty string when an activity has entered the system but is in a queue and has not started executing.

Table 636. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

lock_attributes - Lock attributes monitor element

The lock attributes of the application that is currently holding the lock.

Table 637. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_LOCKS table function - list all locks in the currently connected database	
MON_GET_APPL_LOCKWAIT table function - get information about locks for which an application is waiting	

Table 638. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Lock	lock	Basic
Lock	lock_wait	Basic

Table 639. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks[1]	lock	-
Deadlocks[1]	event_dlconn	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

The following are possible lock attribute settings. Each lock attribute setting is based upon a bit flag value defined in `sqlmon.h`.

Lock Attribute Value in Table Functions	API Constant	Description
0000000000000001	SQLM_LOCKATTR_WAIT_FOR_AVAIL	Wait for availability.
0000000000000002	SQLM_LOCKATTR_ESCALATED	Acquired by escalation.
0000000000000004	SQLM_LOCKATTR_RR_IN_BLOCK	RR lock in block.
0000000000000008	SQLM_LOCKATTR_INSERT	Insert lock.
0000000000000010	SQLM_LOCKATTR_RR	Lock by RR scan.
0000000000000020	SQLM_LOCKATTR_UPDATE_DELETE	Update/delete row lock.
0000000000000040	SQLM_LOCKATTR_ALLOW_NEW	Allow new lock requests.
0000000000000080	SQLM_LOCKATTR_NEW_REQUEST	A new lock requestor.
0000000000000200	SQLM_LOCKATTR_INDOUBT	Lock held by Indoubt Transaction.
0000000000000400	SQLM_LOCKATTR_LOW_PRIORITY	Lock held by low priority application.

Bits returned that are not listed in the table above are reserved for internal use.

lock_count - Lock count monitor element

The number of locks on the lock being held.

Table 640. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Lock	lock	Basic

Table 641. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks[1]	lock	-
Deadlocks[1]	event_dlconn	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

This value ranges from 1 to 255. It is incremented as new locks are acquired, and decremented as locks are released.

When the **lock_count** monitor element has a value of 255, this indicates that a transaction duration lock is being held. At this point, the **lock_count** monitor element is no longer incremented or decremented when locks are acquired or released. The **lock_count** monitor element is set to a value of 255 in one of two possible ways:

1. The **lock_count** monitor element value is incremented 255 times due to new locks being acquired.

2. A transaction duration lock is explicitly acquired. For example, with a LOCK TABLE statement, or an INSERT.

lock_current_mode - Original lock mode before conversion monitor element

During a lock conversion operation, the lock mode held by the application waiting to acquire the lock, before the conversion is completed.

Table 642. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_LOCKS table function - list all locks in the currently connected database	
MON_GET_APPL_LOCKWAIT table function - get information about locks for which an application is waiting	

Table 643. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Lock	lock	Basic
Lock	lock_wait	Basic

Table 644. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks[1]	lock	-
Deadlocks[1]	event_dlconn	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

The following scenario describes an example of lock conversion. During an update or delete operation it is possible to wait for an X lock on the target row. If the transaction is holding an S or V lock on the row, this would require a conversion. At this point, the **lock_current_mode** element is assigned a value of S or V, while the lock waits to be converted to an X lock.

The possible lock modes are listed in the following table.

Mode	Type of Lock	API Constant
	No Lock	SQLM_LNON
IS	Intention Share Lock	SQLM_LOIS
IX	Intention Exclusive Lock	SQLM_LOIX
S	Share Lock	SQLM_LOOS

Mode	Type of Lock	API Constant
SIX	Share with Intention Exclusive Lock	SQLM_LSIX
X	Exclusive Lock	SQLM_LOOX
IN	Intent None	SQLM_LOIN
Z	Super Exclusive Lock	SQLM_LOOZ
U	Update Lock	SQLM_LOOU
NS	Scan Share Lock	SQLM_LONS
NW	Next Key Weak Exclusive Lock	SQLM_LONW

lock_escalation - Lock escalation monitor element

Indicates whether the application waiting to acquire this lock was a result of a lock escalation request. The possible values are Y (Yes) and N (No).

Table 645. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_APPL_LOCKWAIT table function - get information about locks for which an application is waiting	

Table 646. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Lock	lock	Lock
Lock	lock_wait	Lock

Table 647. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks[1]	lock	-
Deadlocks[1]	event_dlconn	-
Deadlocks with Details[1]	event_detailed_dlconn	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

Use this element to better understand the cause of deadlocks. If you experience a deadlock that involves applications doing lock escalation, you may want to increase the amount of lock memory or change the percentage of locks that any one application can request.

lock_escals - Number of lock escalations monitor element

The number of times that locks have been escalated from several row locks to a table lock.

Table 648. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 649. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 650. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE

Table 650. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Database	event_db	-
Connection	event_conn	-
Transactions	event_xact	-
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

A lock is escalated when the total number of locks held by an application reaches the maximum amount of lock list space available to the application, or the lock list space consumed by all applications is approaching the total lock list space. The amount of lock list space available is determined by the **maxlocks** and **locklist** configuration parameters.

When an application reaches the maximum number of locks allowed and there are no more locks to escalate, it will then use space in the lock list allocated for other applications. When the entire lock list is full, an error occurs.

This data item includes a count of all lock escalations, including exclusive lock escalations.

There are several possible causes for excessive lock escalations:

- The lock list size (**locklist**) may be too small for the number of concurrent applications
- The percent of the lock list usable by each application (**maxlocks**) may be too small
- One or more applications may be using an excessive number of locks.

To resolve these problems, you may be able to:

- Increase the **locklist** configuration parameter value.
- Increase the **maxlocks** configuration parameter value.
- Identify the applications with large numbers of locks, or those that are holding too much of the lock list, using one of the following formulae, and comparing the value to **maxlocks**.
  - On 64-bit systems, `(((locks held * 64) / (locklist * 4096)) * 100)`
  - On 32-bit systems, `(((locks held * 48) / (locklist * 4096)) * 100)`

These applications can also cause lock escalations in other applications by using too large a portion of the lock list. These applications may need to resort to using table locks instead of row locks, although table locks may cause an increase in **lock_waits** and **lock_wait_time** monitor element values.

lock_hold_count - Lock hold count monitor element

The number of holds placed on the lock. Holds are placed on locks by cursors registered with the WITH HOLD clause and some DB2 utilities. Locks with holds are not released when transactions are committed.

Table 651. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Lock	lock	Basic

Table 652. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks[1]	lock	-
Deadlocks[1]	event_dlconn	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

lock_list_in_use - Total lock list memory in use monitor element

The total amount of lock list memory (in bytes) that is in use.

Table 653. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Usage

This element may be used in conjunction with the **locklist** configuration parameter to calculate the lock list utilization. If the lock list utilization is high, you may want to consider increasing the size of that parameter.

Note: When calculating utilization, it is important to note that the **locklist** configuration parameter is allocated in pages of 4 KB each, while this monitor element provides results in bytes.

lock_mode - Lock mode monitor element

The type of lock being held. If the mode is unknown, the value of this monitor element is NULL.

Table 654. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_LOCKS table function - list all locks in the currently connected database	Always collected
MON_GET_APPL_LOCKWAIT table function - get information about locks for which an application is waiting	Always collected

Table 655. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Lock
Lock	lock	Lock
Lock	lock_wait	Lock

Table 656. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks[1]	lock	-
Deadlocks[1]	event_dlconn	-
Deadlocks with Details[1]	event_detailed_dlconn	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

This mode can help you determine the source of contention for resources.

This element indicates one of the following, depending on the type of monitor information being examined:

- The type of lock another application holds on the object that this application is waiting to lock (for application-monitoring and deadlock-monitoring levels).
- The type of lock held on the object by this application (for object-lock levels).

The possible values for this field are:

Mode	Type of Lock	API Constant
	No Lock	SQLM_LNON
IS	Intention Share Lock	SQLM_LOIS
IX	Intention Exclusive Lock	SQLM_LOIX
S	Share Lock	SQLM_LOOS

Mode	Type of Lock	API Constant
SIX	Share with Intention Exclusive Lock	SQLM_LSIX
X	Exclusive Lock	SQLM_LOOX
IN	Intent None	SQLM_LOIN
Z	Super Exclusive Lock	SQLM_LOOZ
U	Update Lock	SQLM_LOOU
NS	Scan Share Lock	SQLM_LONS
NW	Next Key Weak Exclusive Lock	SQLM_LONW

lock_mode_requested - Lock mode requested monitor element

The mode in which the lock was requested by the application waiting to acquire the lock.

Table 657. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_APPL_LOCKWAIT table function - get information about locks for which an application is waiting	

Table 658. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Lock	lock_wait	Lock

Table 659. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks[1]	event_dlconn	-
Deadlocks with Details[1]	event_detailed_dlconn	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

The mode in which the lock was requested by the application. This value can help you determine the source of contention for resources.

The possible lock modes are listed in the following table.

Mode	Type of Lock	API Constant
	No Lock	SQLM_LNON
IS	Intention Share Lock	SQLM_LOIS

Mode	Type of Lock	API Constant
IX	Intention Exclusive Lock	SQLM_LOIX
S	Share Lock	SQLM_LOOS
SIX	Share with Intention Exclusive Lock	SQLM_LSIX
X	Exclusive Lock	SQLM_LOOX
IN	Intent None	SQLM_LOIN
Z	Super Exclusive Lock	SQLM_LOOZ
U	Update Lock	SQLM_LOOU
NS	Scan Share Lock	SQLM_LONS
NW	Next Key Weak Exclusive Lock	SQLM_LONW

lock_name - Lock name monitor element

Internal binary lock name. This element serves as a unique identifier for locks.

Table 660. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_LOCKS table function - list all locks in the currently connected database	
MON_GET_APPL_LOCKWAIT table function - get information about locks for which an application is waiting	

Table 661. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Lock	lock	Basic
Lock	lock_wait	lock_wait

Table 662. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks[1]	lock	-
Deadlocks[1]	event_dlconn	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

The internal name can be formatted using the routine MON_FORMAT_LOCK_NAME to obtain more details about the lock. For example, if this is a table lock, then you can obtain the table and tablespace that the lock references.

lock_node - Lock Node

The node involved in a lock.

Table 663. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement
Deadlocks	event_dlconn	Statement
Deadlocks with Details	event_detailed_dlconn	Statement

Usage This can be used for troubleshooting.

lock_object_name - Lock Object Name

This element is provided for informational purposes only. It is the name of the object for which the application holds a lock (for object-lock-level information), or the name of the object for which the application is waiting to obtain a lock (for application-level and deadlock-level information).

Note: This monitor element has been deprecated. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

Table 664. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Lock
Lock	appl_lock_list	Lock
Lock	lock	Basic

Table 665. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	lock	-
Deadlocks	event_dlconn	-
Deadlocks with Details	event_detailed_dlconn	-

Usage For table-level locks, it is the file ID (FID) for SMS and DMS table spaces. For row-level locks, the object name is the row ID (RID). For table space locks, the object name is blank. For buffer pool locks, the object name is the name of the buffer pool.

To determine the table holding the lock, use *table_name* and *table_schema* instead of the file ID, since the file ID may not be unique.

To determine the table space holding the lock, use *tablespace_name.*

lock_object_type - Lock object type waited on monitor element

The type of object against which the application holds a lock (for object-lock-level information), or the type of object for which the application is waiting to obtain a lock (for application-level and deadlock-level information).

Table 666. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_FORMAT_LOCK_NAME table function - format the internal lock name and return details	Always collected
MON_GET_APPL_LOCKWAIT table function - get information about locks for which an application is waiting	Always collected
MON_GET_LOCKS table function - list all locks in the currently connected database	Always collected

Table 667. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Lock
Lock	appl_lock_list	Lock
Lock	lock	Basic
Lock	lock_wait	Lock

Table 668. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks[1]	lock	-
Deadlocks[1]	event_dlconn	-
Deadlocks with Details[1]	event_detailed_dlconn	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

This element can help you determine the source of contention for resources.

For snapshot monitoring and the deadlock[1] event monitor, the object type identifiers are defined in `sqlmon.h`. The objects may be one of the following types:

- Table space (SQLM_TABLESPACE_LOCK in sqlmon.h)
- Table
- Buffer pool
- Block
- Record (or row)
- Data partition (SQLM_TABLE_PART_LOCK in sqlmon.h)

- Internal (another type of lock held internally by the database manager)
- Automatic resize
- Automatic storage.

For the locking event monitor and the monitoring table functions in Table 1, the possible values for the **lock_object_type** monitor element are defined in Table 4.

Table 669. Possible values for lock_object_type monitor element

Possible values	Description
TABLE	Table lock
ROW	Row lock
TABLESPACE	Table space lock
EOT	End of table lock
KEYVALUE	Key value lock
SYSBOOT	Sysboot lock
PLAN	Plan lock
VARIATION	Variation lock
SEQUENCE	Sequence lock
BUFFERPOOL	Buffer pool lock
LOB	LOB/Long region lock
CATALOG	Catalog cache lock
ONLINE_BACKUP	Online backup lock
OBJECT_TABLE	Object table lock
ALTER_TABLE	Table alter lock
DMS_SEQUENCE	DMS sequence lock
REORG	Inplace reorganization lock
MDC_BLOCK	MDC block lock
TABLE_PARTITION	Table partition lock
AUTORESIZE	Autoresize lock
AUTOSTORAGE	Autostorage lock
XMLPATH	XML path lock
EXTENT_MOVEMENT	Extent movement lock
WORKLOAD	Workload authorization lock
FED_SERVER	Federation server lock
FED_USER	Federation user mapping lock
CHUNK	Chunk lock
LOAD_PRE_PART	Load table pre-partitioning lock
LOAD_PART	Load table partitioning lock
LOAD_TS	Loading table space lock
LONG_FIELD_ESC	Long field escalation lock
LONG_FIELD_SPACE	Long field buddy space lock

lock_release_flags - Lock release flags monitor element

Lock release flags.

Table 670. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Lock	lock	Basic
Lock	lock_wait	Basic

Table 671. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks[1]	lock	-
Deadlocks[1]	event_dlconn	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

The following are possible release flag settings. Each release flag is based upon a bit flag value defined in `sqlmon.h`.

API Constant	Description
SQLM_LOCKRELFLAGS_SQLCOMPILER	Locks by SQL compiler.
SQLM_LOCKRELFLAGS_UNTRACKED	Non-unique, untracked locks.

Note: All non-assigned bits are used for application cursors.

lock_status - Lock status monitor element

Indicates the internal status of the lock.

Table 672. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_LOCKS table function - list all locks in the currently connected database	Always collected
MON_GET_APPL_LOCKWAIT table function - get information about locks for which an application is waiting	Always collected

Table 673. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Lock	lock	Basic

Table 674. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks[1]	lock	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

This element can help explain what is happening when an application is waiting to obtain a lock on an object. While it may appear that the application already has a lock on the object it needs, it may have to wait to obtain a different type of lock on the same object.

The lock can be in one of the following statuses:

G Granted state: The application has the lock in the state specified by the **lock_mode** monitor element.

C Converting state: The application is trying to change the lock held to a different type; for example, changing from a share lock to an exclusive lock.

W Waiting state.

Note: API users should refer to the `sqlmon.h` header file containing definitions of database system monitor constants.

lock_timeout_val - Lock timeout value monitor element

Indicates the timeout value (in seconds) when an application has issued a SET CURRENT LOCK TIMEOUT statement. In cases where the statement has not been executed, the database level lock timeout will be shown.

Table 675. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic
Application	agent	Basic

Table 676. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-

Usage

The SET CURRENT LOCK TIMEOUT statement can be used to specify the maximum duration for which application agents will wait for a table or index lock.

If an application is waiting too long on a lock, you can check the **lock_timeout_val** monitor element value to see whether it is set too high inside the application. You

can modify the application to lower the lock timeout value to let the application time out, if that is appropriate for the application logic. You can accomplish this modification with the SET CURRENT LOCK TIMEOUT statement.

If the application is timing out frequently, you can check whether the lock timeout value is set too low and increase it as appropriate.

lock_timeouts - Number of lock timeouts monitor element

The number of times that a request to lock an object timed out instead of being granted.

Table 677. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 678. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 679. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Database	event_db	-
Connection	event_conn	-
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

This element can help you adjust the setting for the **locktimeout** database configuration parameter. If the number of lock timeouts becomes excessive when compared to normal operating levels, you may have an application that is holding locks for long durations. In this case, this element may indicate that you should analyze some of the other lock and deadlock monitor elements to determine if you have an application problem.

You could also have too few lock timeouts if your **locktimeout** database configuration parameter is set too high. In this case, your applications may wait excessively to obtain a lock.

lock_wait_end_time - Lock wait end timestamp monitor element

The date and time that this application stopped waiting to obtain a lock on the object that was locked by another application.

Table 680. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-

Usage

This element can help you determine the severity of resource contention.

lock_wait_start_time - Lock wait start timestamp monitor element

The date and time that this application started waiting to obtain a lock on the object that is currently locked by another application.

Table 681. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_APPL_LOCKWAIT table function - get information about locks for which an application is waiting	Always collected

Table 682. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Lock, Timestamp
Lock	lock_wait	Lock, Timestamp

Table 683. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks [1]	event_dlconn	Timestamp
Deadlocks with Details [1]	event_detailed_dlconn	Timestamp

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

This element can help you determine the severity of resource contention.

lock_wait_time - Time waited on locks monitor element

The total elapsed time spent waiting for locks. The value is given in milliseconds.

Table 684. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 684. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 685. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Lock
Application	appl	Lock
Lock	appl_lock_list	appl_lock_list

For snapshot monitoring, this counter can be reset.

Table 686. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Database	event_db	-
Connection	event_conn	-
Transactions	event_xact	-
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

Table 686. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.		

Usage Notes

At the database level, this is the total amount of elapsed time that all applications were waiting for a lock within this database. This measure of elapsed time can include time spent on locks taken during activities, as well as locks taken during other processing, such compilation.

At the application-connection and transaction levels, this is the total amount of elapsed time that this connection or transaction has waited for a lock to be granted to it.

The value for this element does not include lock wait times for agents that are currently still in a lock wait state. It only includes lock wait times for agents that have already completed their lock waits.

This element may be used in conjunction with the **lock_waits** monitor element to calculate the average wait time for a lock. This calculation can be performed at either the database or the application-connection level.

When using monitor elements providing elapsed times, you should consider:

- Elapsed times are affected by system load, so the more processes you have running, the higher this elapsed time value.
- To calculate this element at the database level, the database system monitor sums the application-level times. This can result in double counting elapsed times at a database level, since more than one application process can be running at the same time.

  To provide meaningful data, you can calculate the average wait time for a lock, as described above.

lock_wait_time_top – Lock wait time top monitor element

The high watermark for lock wait times of any request in a workload. Units are milliseconds. The lock_wait_time_top high watermark is always collected for workloads. A request contributes towards this high watermark only when request metrics are enabled.

Table 687. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wlstats	-

Usage

Use this element to determine the highest lock wait time of any request on a partition for a workload during the time interval collected.

lock_waits - Lock waits monitor element

The total number of times that applications or connections waited for locks.

Table 688. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 689. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 690. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE

Table 690. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Database	event_db	-
Connection	event_conn	-
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

At the database level, this is the total number of times that applications have had to wait for locks within this database.

At the application-connection level, this is the total number of times that this connection requested a lock but had to wait because another connection was already holding a lock on the data.

This element may be used with **lock_wait_time** to calculate, at the database level, the average wait time for a lock. This calculation can be done at either the database or the application-connection level.

If the average lock wait time is high, you should look for applications that hold many locks, or have lock escalations, with a focus on tuning your applications to improve concurrency, if appropriate. If escalations are the reason for a high average lock wait time, then the values of one or both of the **locklist** and **maxlocks** configuration parameters may be too low.

locks_held - Locks held monitor element

The number of locks currently held.

Table 691. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	Always collected
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected

Table 691. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected

Table 692. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic
Lock	db_lock_list	Basic
Lock	appl_lock_list	Basic

Table 693. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks with Details	event_detailed_dlconn	-

Usage

If the monitor information is at the database level, this is the total number of locks currently held by all applications in the database.

If the monitor information is at the application level, this is the total number of locks currently held by all agents for the application.

locks_held_top - Maximum number of locks held monitor element

The maximum number of locks held during this transaction.

Table 694. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Transactions	event_xact	-

Usage

You can use this element to determine if your application is approaching the maximum number of locks available to it, as defined by the **maxlocks** configuration parameter. This parameter indicates the percentage of the lock list that each application can use before lock escalations occur. Lock escalations can result in a decrease in concurrency between applications connected to a database.

Since the **maxlocks** parameter is specified as a percentage and this element is a counter, you can compare the count provided by this element against the total number of locks that can be held by an application, as calculated using one of the following formulae:

- On 64-bit systems, `(locklist * 4096 / 64 ) * (maxlocks / 100)`
- On 32-bit systems, `(locklist * 4096 / 48 ) * (maxlocks / 100)`

If you have a large number of locks, you may need to perform more commits within your application so that some of the locks can be released.

locks_in_list - Number of Locks Reported

The number of locks held by a particular application to be reported on by the event monitor.

Table 695. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks with Details	event_detailed_dlconn	-

locks_waiting - Current agents waiting on locks monitor element

Indicates the number of agents waiting on a lock.

Table 696. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic
Lock	db_lock_list	Basic

Usage

When used in conjunction with **appls_cur_cons**, this element indicates the percentage of applications waiting on locks. If this number is high, the applications may have concurrency problems, and you should identify applications that are holding locks or exclusive locks for long periods of time.

log_buffer_wait_time - Log buffer wait time monitor element

The amount of time an agent spends waiting for space in the log buffer. The value is given in milliseconds.

Table 697. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 697. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 698. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

log_disk_wait_time - Log disk wait time monitor element

The amount of time an agent spends waiting for log records to be flushed to disk. The value is given in milliseconds.

Table 699. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE

Table 699. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 700. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

log_disk_waits_total - Total log disk waits monitor element

The number of times agents have to wait for log data to write to disk.

Table 701. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 702. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE

Table 702. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

log_held_by_dirty_pages - Amount of Log Space Accounted for by Dirty Pages

The amount of log (in bytes) corresponding to the difference between the oldest dirty page in the database and the top of the active log.

Element identifier
log_held_by_dirty_pages

Element type
watermark

Table 703. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Table 704. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage When the snapshot is taken, this value is calculated based on conditions at the time of that snapshot.

Use this element to evaluate the effectiveness of page cleaning for older pages in the buffer pool.

The cleaning of old pages in the buffer pool is governed by the *softmax* database configuration parameter. If the page cleaning is effective then *log_held_by_dirty_pages* should be less than or approximately equal to:

```
(softmax / 100)  *  logfilsiz * 4096
```

If this statement is not true, increase the number of page cleaners (*num_iocleaners*) configuration parameter.

If the condition is true and it is desired that less log be held by dirty pages, then decrease the *softmax* configuration parameter.

log_read_time - Log Read Time

The total elapsed time spent by the logger reading log data from the disk. For event monitors that write to tables, the value of this element is given in microseconds by using the BIGINT data type.

Table 705. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

For snapshot monitoring, this counter can be reset.

Table 706. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage Use this element in conjunction with the *log_reads*, *num_log_read_io*, and *num_log_data_found_in_buffer* elements to determine if:

- The current disk is adequate for logging.
- The log buffer size is adequate.

log_reads - Number of Log Pages Read

The number of log pages read from disk by the logger.

Element identifier
log_reads

Element type
counter

Table 707. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

For snapshot monitoring, this counter can be reset.

Table 708. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage You can use this element with an operating system monitor to quantify the amount of I/O on a device that is attributable to database activity.

log_to_redo_for_recovery - Amount of Log to be Redone for Recovery

The amount of log (in bytes) that will have to be redone for crash recovery.

Element identifier
log_to_redo_for_recovery

Element type
watermark

Table 709. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Table 710. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage When the snapshot is taken, this value is calculated based on conditions at the time of that snapshot. Larger values indicate longer recovery times after a system crash. If the value seems excessive, check the *log_held_by_dirty_pages* monitor element to see if page cleaning needs to be tuned. Also check if there are any long running transactions that need to be terminated.

log_write_time - Log Write Time

The total elapsed time spent by the logger writing log data to the disk. For event monitors that write to tables, the value of this element is given in microseconds by using the BIGINT data type.

Table 711. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

For snapshot monitoring, this counter can be reset.

Table 712. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage Use this element in conjunction with the *log_writes* and *num_log_write_io* elements to determine if the current disk is adequate for logging.

log_writes - Number of Log Pages Written

The number of log pages written to disk by the logger.

Element identifier
log_writes

Element type
counter

Table 713. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

For snapshot monitoring, this counter can be reset.

Table 714. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage You may use this element with an operating system monitor to quantify the amount of I/O on a device that is attributable to database activity.

Note: When log pages are written to disk, the last page might not be full. In such cases, the partial log page remains in the log buffer, and additional log records are written to the page. Therefore log pages might be written to disk by the logger more than once. You should not use this element to measure the number of pages produced by DB2.

long_object_pages - Long Object Pages

The number of disk pages consumed by long data in a table.

Table 715. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table	Basic

Table 716. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Tables	event_table	-

Usage This element provides a mechanism for viewing the actual amount of space consumed by long data in a particular table. This element can be used in conjunction with a table event monitor to track the rate of long data growth over time.

long_object_l_pages - Long object data logical pages monitor element

The number of logical pages used on disk by long data contained in this table.

Table 717. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLE table function - Get table metrics	DATA OBJECT METRICS BASE

Usage

- This value might be less than the amount of space physically allocated for the object. This can happen when you use the REUSE STORAGE option of the TRUNCATE statement. This option causes storage allocated for the table to continue to be allocated, although the storage will be considered empty. In addition, the value for this monitor element might be less than the amount of space logically allocated for the object, because the total space logically allocated includes a small amount of additional meta data.

  To retrieve an accurate measure of the logical or physical size of an object, use the ADMIN_GET_TAB_INFO_V97 function. This function provides more accurate information about the size of objects than you can obtain by multiplying the number of pages reported for this monitor element by the page size.

long_tbsp_id - Long table space ID monitor element

An identifier of the table space that holds long data (LONG or LOB type columns) for this table.

Table 718. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLE table function - Get table metrics	Always collected

Usage

The value of this element matches a value from column TBSPACEID of view SYSCAT.TABLESPACES.

machine_identification - Host hardware identification monitor element

A string that describes the processor architecture. For example, "x86 64 bit". The value returned for this identifier is determined by the operating system running on the host.

Table 719. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

max_agent_overflows - Maximum Agent Overflows

The number of times a request to create a new agent was received when the Maximum Number of Agents (**maxagents**) configuration parameter had already been reached.

Note: The **max_agent_overflows** monitor element is deprecated starting with DB2 Version 9.5. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

Table 720. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

Usage

If agent creation requests are still being received when the **maxagents** configuration parameter has been reached, this might indicate too high a workload for this node.

max_data_received_1024 - Number of Statements with Outbound Bytes Received Between 513 and 1024 Bytes

This element represents the number of statements or chains with outbound bytes received between 513 and 1024 inclusive.

Element identifier
max_data_received_1024

Element type
counter

Table 721. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_data_received_128 - Number of Statements with Outbound Bytes Received Between 1 and 128 Bytes

This element represents the number of statements or chains with outbound bytes received between 1 and 128 inclusive.

Element identifier
max_data_received_128

Element type
counter

Table 722. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_data_received_16384 - Number of Statements with Outbound Bytes Received Between 8193 and 16384 Bytes

This element represents the number of statements or chains with outbound bytes received between 8193 and 16384 inclusive.

Element identifier
max_data_received_16384

Element type
counter

Table 723. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_data_received_2048 - Number of Statements with Outbound Bytes Received Between 1025 and 2048 Bytes

This element represents the number of statements or chains with outbound bytes received between 1025 and 2048 inclusive.

Element identifier
max_data_received_2048

Element type
counter

Table 724. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_data_received_256 - Number of Statements with Outbound Bytes Received Between 129 and 256 Bytes

This element represents the number of statements or chains with outbound bytes received between 129 and 256 inclusive.

Element identifier
max_data_received_256

Element type
counter

Table 725. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_data_received_31999 - Number of Statements with Outbound Bytes Received Between 16385 and 31999 Bytes monitor element

This element represents the number of statements or chains with outbound bytes received between 16385 and 31999 inclusive.

Table 726. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_data_received_4096 - Number of Statements with Outbound Bytes Received Between 2049 and 4096 Bytes

This element represents the number of statements or chains with outbound bytes received between 2049 and 4096 inclusive.

Element identifier
max_data_received_4096

Element type
counter

Table 727. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_data_received_512 - Number of Statements with Outbound Bytes Received Between 257 and 512 Bytes

This element represents the number of statements or chains with outbound bytes received between 257 and 512 inclusive.

Element identifier
max_data_received_512

Element type
counter

Table 728. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_data_received_64000 - Number of Statements with Outbound Bytes Received Between 32000 and 64000 Bytes monitor element

This element represents the number of statements or chains with outbound bytes received between 32000 and 64000 inclusive.

Table 729. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_data_received_8192 - Number of Statements with Outbound Bytes Received Between 4097 and 8192 Bytes

This element represents the number of statements or chains with outbound bytes received between 4097 and 8192 inclusive.

Element identifier
max_data_received_8192

Element type
counter

Table 730. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_data_received_gt64000 - Number of Statements with Outbound Bytes Received Greater than 64000 Bytes

This element represents the number of statements or chains with outbound bytes received greater than 64000.

Element identifier
max_data_received_gt64000

Element type
counter

Table 731. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_data_sent_1024 - Number of Statements with Outbound Bytes Sent Between 513 and 1024 Bytes

This element represents the number of statements or chains with outbound bytes sent between 513 and 1024 inclusive.

Element identifier
max_data_sent_1024

Element type
counter

Table 732. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_data_sent_128 - Number of Statements with Outbound Bytes Sent Between 1 and 128 Bytes

This element represents the number of statements or chains with outbound bytes sent between 1 and 128 inclusive.

Element identifier
max_data_sent_128

Element type
counter

Table 733. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_data_sent_16384 - Number of Statements with Outbound Bytes Sent Between 8193 and 16384 Bytes

This element represents the number of statements or chains with outbound bytes sent between 8193 and 16384 inclusive.

Element identifier
max_data_sent_16384

Element type
counter

Table 734. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_data_sent_2048 - Number of Statements with Outbound Bytes Sent Between 1025 and 2048 Bytes

This element represents the number of statements or chains with outbound bytes sent between 1025 and 2048 inclusive.

Element identifier
max_data_sent_2048

Element type
counter

Table 735. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement

Table 735. Snapshot Monitoring Information (continued)

Snapshot Level	Logical Data Grouping	Monitor Switch
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_data_sent_256 - Number of Statements with Outbound Bytes Sent Between 129 and 256 Bytes

This element represents the number of statements or chains with outbound bytes sent between 129 and 256 inclusive.

Element identifier
max_data_sent_256

Element type
counter

Table 736. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_data_sent_31999 - Number of Statements with Outbound Bytes Sent Between 16385 and 31999 Bytes

This element represents the number of statements or chains with outbound bytes sent between 16385 and 31999 inclusive.

Element identifier
max_data_sent_31999

Element type
counter

Table 737. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_data_sent_4096 - Number of Statements with Outbound Bytes Sent Between 2049 and 4096 Bytes

This element represents the number of statements or chains with outbound bytes sent between 2049 and 4096 inclusive.

Element identifier
max_data_sent_4096

Element type
counter

Table 738. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_data_sent_512 - Number of Statements with Outbound Bytes Sent Between 257 and 512 Bytes

This element represents the number of statements or chains with outbound bytes sent between 257 and 512 inclusive.

Element identifier
max_data_sent_512

Element type
counter

Table 739. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_data_sent_64000 - Number of Statements with Outbound Bytes Sent Between 32000 and 64000 Bytes

This element represents the number of statements or chains with outbound bytes sent between 32000 and 64000 inclusive.

Element identifier
max_data_sent_64000

Element type
counter

Table 740. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_data_sent_8192 - Number of Statements with Outbound Bytes Sent Between 4097 and 8192 Bytes

This element represents the number of statements or chains with outbound bytes sent between 4097 and 8192 inclusive.

Element identifier
max_data_sent_8192

Element type
counter

Table 741. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_data_sent_gt64000 - Number of Statements with Outbound Bytes Sent Greater than 64000 Bytes

This element represents the number of statements or chains with outbound bytes sent greater than 64000.

Element identifier
max_data_sent_gt64000

Element type
counter

Table 742. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_network_time_100_ms - Number of Statements with Network Time between 16 and 100 ms

This element represents the number of statements or chains whose network time was greater than 16 milliseconds but less or equal to 100 milliseconds. (Network time is the difference between host response time and elapsed execution time for a statement or chain.)

Table 743. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_network_time_16_ms - Number of Statements with Network Time between 4 and 16 ms

This element represents the number of statements or chains whose network time was greater than 4 milliseconds but less or equal to 16 milliseconds. (Network time is the difference between host response time and elapsed execution time for a statement or chain.)

Element identifier
max_network_time_16_ms

Element type
counter

Table 744. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_network_time_1_ms - Number of Statements with Network Time of up to 1 ms

This element represents the number of statements or chains whose network time was less or equal to 1 millisecond. (Network time is the difference between host response time and elapsed execution time for a statement or chain.)

Table 745. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_network_time_4_ms - Number of Statements with Network Time between 1 and 4 ms

This element represents the number of statements or chains whose network time was greater than 1 millisecond but less or equal to 4 milliseconds. (Network time is the difference between host response time and elapsed execution time for a statement or chain.)

Element identifier
max_network_time_4_ms

Element type
counter

Table 746. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_network_time_500_ms - Number of Statements with Network Time between 100 and 500 ms

This element represents the number of statements or chains whose network time was greater than 100 milliseconds but less or equal to 500 milliseconds. (Network time is the difference between host response time and elapsed execution time for a statement or chain.)

Table 747. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

max_network_time_gt500_ms - Number of Statements with Network Time greater than 500 ms

This element represents the number of statements or chains whose network time was greater than 500 milliseconds. (Network time is the difference between host response time and elapsed execution time for a statement or chain.)

Table 748. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

Usage Use this element to get a better idea of the database activity and network traffic at the database or application levels.

member - Database member monitor element

The numeric identifier for the database member from which the data was retrieved for this result record.

Table 749. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_DB2_SYSTEM_RESOURCES table function - Return DB2(r) system information	Always collected
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected
ENV_GET_NETWORK_RESOURCES table function - Return network adapter information	Always collected
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details	Always collected
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	Always collected
MON_GET_CONNECTION table function - Get connection metrics	Always collected

Table 749. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics	Always collected
MON_GET_CONTAINER table function - Get table space container metrics	Always collected
MON_GET_EXTENT_MOVEMENT_STATUS - Get extent movement progress status metrics	Always collected
MON_GET_FCM - Get FCM metrics	Always collected
MON_GET_FCM_CONNECTION_LIST - Get details for all FCM connections	Always collected
MON_GET_INDEX table function - Get index metrics	Always collected
MON_GET_LOCKS table function - list all locks in the currently connected database	
MON_GET_MEMORY_POOL table function - Get memory pool information	Always collected
MON_GET_MEMORY_SET table function - Get memory set information	Always collected
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	Always collected
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	Always collected
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics	Always collected
MON_GET_TABLE table function - Get table metrics	Always collected
MON_GET_TABLESPACE table function - Get table space metrics	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics	Always collected
MON_GET_WORKLOAD table function - Get workload metrics	Always collected
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics	Always collected

Table 750. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	-	ACTIVITY METRICS BASE
Statistics	-	REQUEST METRICS BASE

Table 750. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	-	REQUEST METRICS BASE
Unit of work	-	REQUEST METRICS BASE
Package cache	-	COLLECT BASE DATA
Locking	-	-

Usage

A DB2 member is a database manager instance that runs DB2 server software on a single host. A DB2 member accepts and processes database requests from applications connected to it.

memory_free - Amount of free physical memory monitor element

The total amount of physical memory on this host that is not allocated to a running process, in MB.

Table 751. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

memory_pool_id - Memory pool identifier monitor element

Memory pool identifier

Table 752. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_MEMORY_POOL table function - Get memory pool information	Always collected

memory_pool_type - Memory pool name monitor element

The name of the memory pool.

Table 753. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_MEMORY_POOL table function - Get memory pool information	Always collected

Usage

Use the memory_pool_type element to identify the type of memory pool. The possible values returned by this monitor element are listed in Table 754 on page 763.

Table 754. Possible values returned for memory_pool_type

Memory pool name*	Description	Additional information
APM	Agent pool management (APM) heap	Internal memory pool
APPL_SHARED	Application shared heap	Internal memory pool
APPLICATION	Application heap	See applheapsz - Application heap size configuration parameter.
APS	APS heap	Internal memory pool
BSU_CF	Base service utility (BSU) CF heap	Internal memory pool
BSU	Base service utility (BSU) heap	Internal memory pool
BP	Buffer pool heap	See CREATE BUFFERPOOL statement.
CAT_CACHE	Catalog cache heap	See catalogcache_sz - Catalog cache size configuration parameter.
DATABASE_CF	Database CF heap	Internal memory pool
DATABASE	Database heap	See dbheap - Database heap configuration parameter.
DEBUG	Debug heap	Internal memory pool
DROP_INDEX	Drop index heap	Internal memory pool
EDU	Engine dispatchable unit (EDU) heap	Internal memory pool
FCMBP	Fast communications manager (FCM) buffer heap	See fcm_num_buffers - Number of FCM buffers configuration parameter.
FCM_CHANNEL	FCM channel heap	See fcm_num_channels - Number of FCM channels configuration parameter
FCM_CONTROL	FCM control heap	Internal memory pool
FCM_LOCAL	FCM local heap	Internal memory pool
FCM_SESSION	FCM session heap	Internal memory pool
FEDERATED	Federated heap	Internal memory pool
KERNEL _CONTROL	Kernel control block heap	Internal memory pool
KERNEL	Kernel heap	Internal memory pool
LOCK_MGR	Lock manager heap	See locklist - Maximum storage for lock list configuration parameter.
MISC	Miscellaneous heap	See DB2_FMP_COMM_HEAPSZ registry variable.
MONITOR	Monitor heap	See mon_heap_sz - Database system monitor heap size configuration parameter.
OPTPROF _PARSER	OptProf XML parser heap	Internal memory pool

Table 754. Possible values returned for memory_pool_type (continued)

Memory pool name*	Description	Additional information
OSS_TRACKER	OSS resource tracking heap	Internal memory pool
PERSISTENT _PRIVATE	Persistent private heap	Internal memory pool
PACKAGE _CACHE	Package cache heap	See pckcachesz - Package cache size configuration parameter.
PRIVATE	Private	Internal memory pool
RESYNC	Resync heap	Internal memory pool
SORT	Private sort heap	See sortheap - Sort heap size configuration parameter.
SHARED_SORT	Shared sort heap	See sheapthres_shr - Sort heap threshold for shared sorts configuration parameter.
SQL_COMPILER	SQL compiler heap	Internal memory pool
STATEMENT	Statement heap	See stmtheap - Statement heap size configuration parameter.
STATISTICS	Statistics heap	See stat_heap_sz - Statistics heap size configuration parameter.
USER_DATA	User data heap	Internal memory pool
UTILITY	Utility heap	See util_heap_sz - Utility heap size configuration parameter.
XMLCACHE	XML cache heap	Internal memory pool
XMLPARSER	XML parser heap	Internal memory pool

* Names returned by **db2pd** for these pools might be abbreviated. The names of the pools and any abbreviations used by **db2pd** are defined in sqlopoolinfo.h.

memory_pool_used - Amount of memory pool in use monitor element

Amount of committed memory in use by this memory pool, in KB.

Table 755. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_MEMORY_POOL table function - Get memory pool information	Always collected

memory_pool_used_hwm - Memory pool high water mark monitor element

The highest amount of memory assigned to this pool since it was created, in KB.

Table 756. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_MEMORY_POOL table function - Get memory set infromation	Always collected

memory_set_committed - Memory currently committed monitor element

The amount of memory currently committed to this memory set, in KB.

Table 757. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_MEMORY_SET table function - Get memory set information	Always collected

Usage

Committed memory is memory that is backed by RAM, or paging space, or both on the system.

memory_set_id - Memory set identifier monitor element

A numeric identifier that maps to a specific memory set type.

Table 758. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_MEMORY_SET table function - Get memory set information	Always collected

memory_set_size - Memory set size monitor element

Maximum memory commitment limit, in KB.

This value represents either a configured setting for a memory set or an internally calculated value for those memory sets that are managed automatically.

Table 759. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_MEMORY_SET table function - Get memory set information	Always collected

memory_set_type - Memory set type monitor element

The type of memory set.

Table 760. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_MEMORY_POOL table function - Get memory pool information	Always collected
MON_GET_MEMORY_SET table function - Get memory set information	Always collected

Usage

The possible values returned for this monitor element are described in Table 761 on page 766:

Table 761. Possible values for **memory_set_type**

Memory set type	Description	Scope
DBMS	Database manager memory set	Instance
FMP	Fenced mode process memory set	Instance
PRIVATE	Private memory set	Instance
DATABASE	Database memory set	Database
APPLICATION	Application memory set	Database
FCM	Fast communication manager (FCM) memory set	Instance, Host

memory_set_used - Memory in use by this set monitor element

Amount of memory from this set that has been assigned to memory pools, in KB.

Table 762. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_MEMORY_SET table function - Get memory set information	Always collected

Usage

All of memory represented by this monitor element is committed memory; the value returned for this monitor element is included in MEMORY_SET_COMMITTED. Any additional memory that is committed but not in use is cached to improve performance.

memory_set_used_hwm - Memory set high water mark monitor element

The highest amount of memory assigned to memory pools from this set since the memory set was created, in KB.

Table 763. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_MEMORY_SET table function - Get memory set information	Always collected

memory_swap_free - Total free swap space monitor element

The total amount of unused swap space on this host, in MB.

Table 764. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

memory_swap_total - Total swap space monitor element

The total amount of swap space on this host, in MB.

Table 765. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

memory_total - Total physical memory monitor element

The total amount of physical memory on this host, in MB.

Table 766. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

message - Control Table Message

The nature of the timestamp in the MESSAGE_TIME column. This element is only used in the CONTROL table by write-to-table event monitors.

Table 767. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
-	-	-

Usage

The following are possible values:

DROPPED RECORDS: *n*
Number of activity records that were dropped because MONHEAP could not be allocated for them.

FIRST_CONNECT
The time of the first connect to the database after activation.

EVMON_START
The time the event monitor listed in the EVMONNAME column was started.

OVERFLOWS: *n*
Denotes that *n* records were discarded due to buffer overflow.

LAST DROPPED RECORD
The last time that an activity record was dropped.

message_time - Timestamp Control Table Message

The timestamp corresponding to the event described in the MESSAGE column. This element is only used in the CONTROL table by write-to-table event monitors.

Table 768. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
-	-	-

nesting_level - Nesting level monitor element

This element shows the level of nesting or recursion in effect when the statement was being run; each level of nesting corresponds to nested or recursive invocation of a stored procedure or user-defined function (UDF).

The **nesting_level** monitor element is an alias of the **stmt_nest_level** monitor element.

Table 769. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 770. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values[1]	event_stmt_history	-
Deadlocks with Details History[1]	event_stmt_history	-
Activities	event_activitystmt	-
Unit of work	Reported in the package list.	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

You can use this element, along with **stmt_invocation_id** monitor element, to uniquely identify the invocation in which a particular SQL statement has been executed. You can also use this element in conjunction with other statement history entries to see the sequence of SQL statements that caused the deadlock.

network_time_bottom - Minimum Network Time for Statement

This element represents the shortest network time for a statement executed against this DCS database or in this DCS application, or having used this many data transmissions. (Network time is the difference between host response time and elapsed execution time for a statement.)

Element identifier
network_time_bottom

Element type
watermark

Table 771. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement, Timestamp
DCS Application	dcs_appl	Statement, Timestamp
Data Transmission	stmt_transmissions	Statement, Timestamp

For snapshot monitoring, this counter can be reset.

Usage

Use this element to get a better idea of the database activity and network traffic at the database or application levels.

This element is composed of two subelements that report time spent as seconds and microseconds (one millionth of a second). The names of the subelements can be derived by adding "_s" and "_ms" to the name of this monitor element. To retrieve the total time spent for this monitor element, the values of the two subelements must be added together. For example, if the "_s" subelement value is 3 and the "_ms" subelement value is 20, then the total time spent for the monitor element is 3.00002 seconds.

network_time_top - Maximum Network Time for Statement

This element represents the longest network time for a statement executed against this DCS database or in this DCS application, or having used this many data transmissions. (Network time is the difference between host response time and elapsed execution time for a statement.)

Element identifier
network_time_top

Element type
watermark

Table 772. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement, Timestamp
DCS Application	dcs_appl	Statement, Timestamp
Data Transmission	stmt_transmissions	Statement, Timestamp

For snapshot monitoring, this counter can be reset.

Usage

Use this element to get a better idea of the database activity and network traffic at the database or application levels. Note that this element is not collected when the timestamp switch is off.

This element is composed of two subelements that report time spent as seconds and microseconds (one millionth of a second). The names of the subelements can be derived by adding "_s" and "_ms" to the name of this monitor element. To retrieve the total time spent for this monitor element, the values of the two subelements must be added together. For example, if the "_s" subelement value is 3 and the "_ms" subelement value is 20, then the total time spent for the monitor element is 3.00002 seconds.

nleaf - Number of leaf pages monitor element

The approximate number of leaf pages.

Table 773. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

nlevels - Number of index levels monitor element

Number of index levels. This is an approximation.

Table 774. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

node_number - Node Number

The number assigned to the node in the *db2nodes.cfg* file.

Table 775. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	collected	Basic
Database Manager	memory_pool	Basic
Database Manager	fcm	Basic
Database Manager	fcm_node	Basic
Database Manager	utility_info	Basic
Database	detail_log	Basic
Buffer Pool	bufferpool_nodeinfo	Buffer Pool
Table Space	rollforward	Basic
Lock	lock	Basic
Lock	lock_wait	Basic
Database	db_sto_path_info	Buffer Pool

Table 776. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connections	event_connheader	-
Deadlocks	lock	-

Table 776. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Overflow Record	event_overflow	-
Database	event_dbmemuse	-
Connection	event_connmemuse	-

Usage This value identifies the current node number, which can be used when monitoring multiple nodes.

nonboundary_leaf_node_splits - Non-boundary leaf node splits monitor element

Number of times a non-boundary leaf node was split during an insert operation.

Table 777. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

num_agents - Number of Agents Working on a Statement

Number of concurrent agents currently executing a statement or subsection.

Element identifier
num_agents

Element type
gauge

Table 778. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement
Application	subsection	Statement

Usage An indicator how well the query is parallelized. This is useful for tracking the progress of query execution, by taking successive snapshots.

num_assoc_agents - Number of Associated Agents

At the application level, this is the number of subagents associated with an application. At the database level, it is the number of subagents for all applications.

Element identifier
num_assoc_agents

Element type
gauge

Table 779. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl_info	Basic

Usage You can use this element to help evaluate your settings for your agent configuration parameters.

num_compilations - Statement Compilations

The number of different compilations for a specific SQL statement.

Table 780. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Dynamic SQL	dynsql	Basic

Usage Some SQL statements issued on different schemas, such as "select t1 from foo" will appear to be the same statement in the DB2 cache even though they refer to different access plans. Use this value in conjunction with num_executions to determine whether a bad compilation environment may be skewing the results of dynamic SQL snapshot statistics.

num_coord_exec - Number of executions by coordinator agent monitor element

The number of times this section was executed by a coordinator agent.

Table 781. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	Always collected

Table 782. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Package cache	-	COLLECT BASE DATA

num_coord_exec_with_metrics - Number of executions by coordinator agent with metrics monitor element

The number of times this section was executed by a coordinator agent and monitoring metrics were being captured

Table 783. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	Always collected

Table 784. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Package cache	-	COLLECT BASE DATA

num_db_storage_paths - Number of automatic storage paths

This element shows the number of automatic storage paths associated with this database.

Table 785. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Usage You can use this element with the db_storage_path monitor element to identify the storage paths that are associated with this database.

num_exec_with_metrics - Number of executions with metrics collected monitor element

The number of times that this SQL statement section has been executed with the metrics collected. This element can be used to calculate the per execution value for monitor elements for statements in the package cache.

Table 786. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	Always collected

Table 787. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Package cache	-	COLLECT BASE DATA

num_executions - Statement executions monitor element

The number of times that an SQL statement has been executed.

Table 788. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	Always collected

Table 789. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Dynamic SQL	dynsql	Basic

For snapshot monitoring, this counter can be reset.

Table 790. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Package cache	-	COLLECT BASE DATA

Usage

You can use this element to identify the most frequently executed SQL statements in your system.

At the package cache level, use this element to compute averages for the activity metrics reported per statement. For example, the average CPU usage for an execution of a statement reported at the package cache level can be calculated by the following formula:

```
total_cpu_time / num_exec_with_metrics
```

Use the **num_exec_with_metrics** monitor element instead of the **num_executions** monitor element when computing averages, since the **num_executions** monitor element counts all executions of a statement, regardless of whether or not the execution of the statement contributed to the activity metrics that are reported.

num_extents_left - Number of extents left to process monitor element

The number of extents left to move during this table rebalancing process.

Table 791. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_EXTENT_MOVEMENT_STATUS - Get extent movement progress status metrics	Always collected

num_extents_moved - Number of extents moved monitor element

The number of extents moved so far during this extent movement operation.

Table 792. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_EXTENT_MOVEMENT_STATUS - Get extent movement progress status metrics	Always collected

num_gw_conn_switches - Connection Switches

The number of times that an agent from the agents pool was primed with a connection and was reassigned for use with a different DRDA database.

Table 793. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

Usage

For most users, the default setting of the **num_poolagents** configuration parameter ensures optimal performance. The default setting for this configuration parameter automatically manages agent pooling and avoids reassigning agents.

To reduce the value of this monitor element, adjust the value of the **num_poolagents** configuration parameter.

num_indoubt_trans - Number of Indoubt Transactions

The number of outstanding indoubt transactions in the database.

Table 794. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Usage Indoubt transactions hold log space for uncommitted transactions, which can cause the logs to become full. When the logs are full, further transactions cannot be completed. The resolution of this problem involves a manual process of heuristically resolving the indoubt transactions. This monitor element provides a count of the number of currently outstanding indoubt transactions that must be heuristically resolved.

num_log_buffer_full - Number of times full log buffer caused agents to wait monitor element

The number of times agents had to wait for log data to write to disk while copying log records into the log buffer. This value is incremented per agent per incident. For example, if two agents attempt to copy log data while the buffer is full, then this value is incremented by two.

Table 795. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 795. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 796. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

For snapshot monitoring, this counter can be reset.

Table 797. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

Table 797. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element to determine if the **logbufsz** database configuration parameter needs to be increased.

num_log_data_found_in_buffer - Number of Log Data Found In Buffer

The number of times an agent reads log data from the buffer. Reading log data from the buffer is preferable to reading from the disk because the latter is slower.

Table 798. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

For snapshot monitoring, this counter can be reset.

Table 799. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage Use this element in conjunction with the *num_log_read_io* element to determine if the LOGBUFSZ database configuration parameter needs to be increased.

num_log_part_page_io - Number of Partial Log Page Writes

The number of I/O requests issued by the logger for writing partial log data to the disk.

Table 800. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

For snapshot monitoring, this counter can be reset.

Table 801. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage Use this element in conjunction with the *log_writes*, *log_write_time*, and *num_log_write_io* elements to determine if the current disk is adequate for logging.

num_log_read_io - Number of Log Reads

The number of I/O requests issued by the logger for reading log data from the disk.

Table 802. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

For snapshot monitoring, this counter can be reset.

Table 803. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage Use this element in conjunction with the *log_reads* and *log_read_time* elements to determine if the current disk is adequate for logging.

num_log_write_io - Number of Log Writes

The number of I/O requests issued by the logger for writing log data to the disk.

Table 804. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

For snapshot monitoring, this counter can be reset.

Table 805. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage Use this element in conjunction with the *log_writes* and *log_write_time* elements to determine if the current disk is adequate for logging.

num_lw_thresh_exceeded - Number of lock wait thresholds exceeded monitor element

This monitor element reports the number of times the lock wait threshold (set using **mon_lw_thresh** configuration parameter) was exceeded and a lock wait event was captured by the locking event monitor. If no lock wait event is generated, the monitor element is not incremented.

Table 806. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE

Table 806. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Command and Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 807. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

num_nodes_in_db2_instance - Number of Nodes in Partition

The number of nodes on the instance where the snapshot was taken.

Table 808. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

Table 809. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Event Log Header	event_log_header	-

Usage Use this element to determine the number of nodes for an instance. For non-partitioned system databases, this value will be 1.

num_remaps - Number of remaps monitor element

Count of the number of times this activity has been remapped. If num_remaps is greater than zero, the service_class_id of this activity record is the ID of the last service class to which the activity was remapped.

Table 810. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

Usage

Use this information to verify whether the activity was remapped the expected number of times.

num_threshold_violations - Number of threshold violations monitor element

The number of threshold violations that have taken place in this database since it was last activated.

This monitor element is an alias of the "thresh_violations - Number of threshold violations monitor element" on page 981 monitor element, which is returned by some monitoring (MON_*) table functions.

Table 811. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

For snapshot monitoring, this counter can be reset.

Table 812. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage

This element can be used to help determine whether or not thresholds are effective for this particular application or whether the threshold violations are excessive.

num_transmissions - Number of Transmissions

Number of data transmissions between the DB2 Connect gateway and the host that was used to process this DCS statement. (One data transmission consists of either one send or one receive.)

Note:

This is a legacy monitor element that is not relevant for DB2 UDB Version 8.1.2 or higher. If you are using DB2 UDB Version 8.1.2 or higher, refer to the **num_transmissions_group** monitor element.

Element identifier
num_transmissions

Element type
counter

-->

Table 813. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Statement	dcs_stmt	Statement

Usage Use this element to get a better understanding of the reasons why a particular statement took longer to execute. For example, a query returning a large result set may need many data transmissions to complete.

num_transmissions_group - Number of Transmissions Group

The range of data transmissions between the DB2 Connect gateway and the host that was used to process this DCS statement. (One data transmission consists of either one send or one receive.)

Table 814. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Statement	dcs_stmt	Statement

Usage Use this element to get a better understanding of the reasons why a particular statement took longer to execute. For example, a query returning a large result set may need many data transmissions to complete.

The constants representing the ranges of transmissions are described as follows and are defined in sqlmon.h.

API Constant	Description
SQLM_DCS_TRANS_GROUP_2	2 transmissions
SQLM_DCS_TRANS_GROUP_3TO7	3 to 7 transmissions
SQLM_DCS_TRANS_GROUP_8TO15	8 to 15 transmissions
SQLM_DCS_TRANS_GROUP_16TO64	16 to 64 transmissions

API Constant	Description
SQLM_DCS_TRANS_GROUP_GT64	Greater than 64 transmissions

number_in_bin - Number in bin monitor element

This element holds the count of the number of activities or requests that fall within the histogram bin.

Table 815. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_histogrambin	-

Usage

Use this element to represent the height of a bin in the histogram.

olap_func_overflows - OLAP Function Overflows monitor element

The number of times that OLAP function data exceeded the available sort heap space.

Table 816. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 817. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage

At the database level, use this element in conjunction with total_olap_funcs to calculate the percentage of OLAP functions that overflowed to disk. If this percentage is high and the performance of applications using OLAP functions needs to be improved, then you should consider increasing the sort heap size.

At the application level, use this element to evaluate OLAP function performance for individual applications.

open_cursors - Number of Open Cursors

The number of cursors currently open for an application.

Table 818. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcs_appl	Statement

Usage Use this element to assess how much memory is being allocated. The amount of memory allocated by the DB2 client, DB2 Connect, or the database agent on the target database is related to the number of cursors that are currently open. Knowing this information can help with capacity planning. For example, each open cursor that is doing blocking has a buffer size of `RQRIOBLK`. If *deferred_prepare* is enabled, then two buffers will be allocated.

This element does not include cursors that were closed by an early close. An early close occurs when the host database returns the last record to the client. The cursor is closed at the host and gateway, but is still open at the client. Early close cursors can be set using the DB2 Call Level Interface.

open_loc_curs - Open Local Cursors

The number of local cursors currently open for this application, including those cursors counted by *open_loc_curs_blk.*

Element identifier
open_loc_curs

Element type
gauge

Table 819. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

Usage You may use this element in conjunction with *open_loc_curs_blk* to calculate the percentage of local cursors that are blocking cursors. If the percentage is low, you may be able to improve performance by improving the row blocking in the application.

For cursors used by remote applications, see *open_rem_curs.*

open_loc_curs_blk - Open Local Cursors with Blocking

The number of local blocking cursors currently open for this application.

Element identifier
open_loc_curs_blk

Element type
gauge

Table 820. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

Usage You may use this element in conjunction with *open_loc_curs* to calculate the percentage of local cursors that are blocking cursors. If the percentage is low, you may be able to improve performance by improving the row blocking in the application:

- Check the pre-compile options for record blocking for treatment of ambiguous cursors

- Redefine cursors to allow for blocking (for example, if possible, specify FOR FETCH ONLY on your cursors).

rej_curs_blk and acc_curs_blk provide additional information that may help you tune your configuration parameters to improve row blocking in your application.

For blocking cursors used by remote applications, see *open_rem_curs_blk.*

open_rem_curs - Open Remote Cursors

The number of remote cursors currently open for this application, including those cursors counted by *open_rem_curs_blk.*

Element identifier
open_rem_curs

Element type
gauge

Table 821. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

Usage You may use this element in conjunction with *open_rem_curs_blk* to calculate the percentage of remote cursors that are blocking cursors. If the percentage is low, you may be able to improve performance by improving the row blocking in the application. See *open_rem_curs_blk* for more information.

For the number of open cursors used by applications connected to a local database, see *open_loc_curs.*

open_rem_curs_blk - Open Remote Cursors with Blocking

The number of remote blocking cursors currently open for this application.

Element identifier
open_rem_curs_blk

Element type
gauge

Table 822. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

Usage You can use this element in conjunction with *open_rem_curs* to calculate the percentage of remote cursors that are blocking cursors. If the percentage is low, you may be able to improve performance by improving the row blocking in the application:

- Check the pre-compile options for record blocking for treatment of ambiguous cursors
- Redefine cursors to allow for blocking (for example, if possible, specify FOR FETCH ONLY on your cursors).

rej_curs_blk and *acc_curs_blk* provide additional information that may help you tune your configuration parameters to improve row blocking in your application.

For the number of open blocking cursors used by applications connected to a local database see *open_loc_curs_blk*.

os_level - Operating system level monitor element

The modification level of the operating system running on this host. Reported for Linux systems only.

Table 823. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

os_name - Operating system name monitor element

The name of the operating system running on this host.

Table 824. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

os_release - Operating system release monitor element

The release of the operating system running on this host.

Table 825. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

os_version - Operating system version monitor element

The version of the operating system running on this host.

Table 826. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

outbound_appl_id - Outbound Application ID

This identifier is generated when the application connects to the DRDA host database. It is used to connect the DB2 Connect gateway to the host, while the **appl_id** monitor element is used to connect a client to the DB2 Connect gateway.

Note: NetBIOS is no longer supported. SNA, including its APIs APPC, APPN, and CPI-C, is also no longer supported. If you use these protocols, you must recatalog your nodes and databases using a supported protocol such as TCP/IP. References to these protocols should be ignored.

Element identifier
outbound_appl_id

Element type
information

-->

Table 827. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcs_appl_info	Basic

Usage

You may use this element in conjunction with **appl_id** to correlate the client and server parts of the application information.

This identifier is unique across the network.

This element will be blank when the gateway concentrator is on, or if the DCS application is not in a logical unit of work.

Format
```
Network.LU Name.Application instance
```

Example
```
CAIBMTOR.OSFDBM0.930131194520
```

Details
This application ID is the displayable format of an actual SNA LUWID (Logical Unit-of-Work ID) that flows on the network when an APPC conversation is allocated. APPC-generated application IDs are made up by concatenating the network name, the LU name, and the LUWID instance number, which creates a unique label for the client/server application. The network name and LU name can each be a maximum of 8 characters. The application instance corresponds to the 12-decimal-character LUWID instance number.

outbound_bytes_received - Outbound Number of Bytes Received

The number of bytes received by the DB2 Connect gateway from the host, excluding communication protocol overhead (for example, TCP/IP or SNA headers). For the data transmission level: Number of bytes received by the DB2 Connect gateway from the host during the processing of all the statements that used this number of data transmissions.

Table 828. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Basic
DCS Application	dcs_appl	Basic
DCS Statement	dcs_stmt	Statement

Table 828. Snapshot Monitoring Information (continued)

Snapshot Level	Logical Data Grouping	Monitor Switch
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring at the statement level, this counter cannot be reset. This counter can be reset at other levels.

Usage

Use this element to measure the throughput from the host databases to the DB2 Connect gateway.

outbound_bytes_received_bottom - Minimum Outbound Number of Bytes Received

The lowest number of bytes received per statement or chain by the DB2 Connect gateway from the host during the processing of all the statements or chains against this DCS database, or in this DCS application, that used this number of data transmissions.

Table 829. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Data Transmission	stmt_transmissions	Statement

Usage Use this element in conjunction with "outbound number of bytes received" as yet another parameter that illustrates the throughput from the host database to the DB2 Connect gateway.

outbound_bytes_received_top - Maximum Outbound Number of Bytes Received

Maximum number of bytes received per statement or chain by the DB2 Connect gateway from the host during the processing of all the statements or chains against this DCS database, or in this DCS application, that used this number of data transmissions.

Table 830. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Data Transmission	stmt_transmissions	Statement

Usage Use this element in conjunction with "outbound number of bytes received" as yet another parameter that illustrates the throughput from the host database to the DB2 Connect gateway.

outbound_bytes_sent - Outbound Number of Bytes Sent

The number of bytes sent by the DB2 Connect gateway to the host, excluding communication protocol overhead (for example, TCP/IP or SNA headers). For the data transmission level: Number of bytes sent by the DB2 Connect gateway to the host during the processing of all the statements that used this number of data transmissions.

Table 831. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Basic
DCS Application	dcs_appl	Basic
DCS Statement	dcs_stmt	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring at the statement level, this counter cannot be reset. This counter can be reset at other levels.

Usage Use this element to measure the throughput from the DB2 Connect gateway to the host database.

outbound_bytes_sent_bottom - Minimum Outbound Number of Bytes Sent

The lowest number of bytes sent per statement or chain by the DB2 Connect gateway to the host during the processing of all the statements or chains against this DCS database, or in this DCS application, that used this number of data transmissions.

Table 832. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Data Transmission	stmt_transmissions	Statement

Usage Use this element in conjunction with "outbound number of bytes sent" as yet another parameter that illustrates the throughput from the DB2 Connect Gateway to the host database.

outbound_bytes_sent_top - Maximum Outbound Number of Bytes Sent

Maximum number of bytes sent per statement or chain by the DB2 Connect gateway to the host during the processing of all the statements or chains against this DCS database, or in this DCS application, that used this number of data transmissions.

Table 833. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Data Transmission	stmt_transmissions	Statement

Usage Use this element in conjunction with "outbound number of bytes sent" as yet another parameter that illustrates the throughput from the DB2 Connect Gateway to the host database.

outbound_comm_address - Outbound Communication Address

This is the communication address of the target database. For example, it could be an SNA net ID and LU partner name, or an IP address and port number for TCP/IP.

Element identifier
outbound_comm_address

Element type
information

Table 834. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
DCS Application	dcs_appl_info	Basic

Usage Use this element for problem determination on DCS applications.

outbound_comm_protocol - Outbound Communication Protocol

The communication protocol used between the DB2 Connect gateway and the host.

Table 835. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcs_appl_info	Basic

Usage

Use this element for problem determination on DCS applications. The valid value is:

- SQLM_PROT_TCPIP

outbound_sequence_no - Outbound Sequence Number

This element will be blank when the gateway concentrator is on, or if the DCS application is not in a logical unit of work.

Table 836. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcs_appl_info	Basic

overflow_accesses - Accesses to overflowed records monitor element

The number of accesses (reads and writes) to overflowed rows of this table.

Table 837. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLE table function - Get table metrics	Always collected

Table 838. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table	Basic

For snapshot monitoring, this counter can be reset.

Table 839. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Tables	event_table	-

Usage

Overflowed rows indicate that data fragmentation has occurred. If this number is high, you may be able to improve table performance by reorganizing the table using the **REORG** utility, which cleans up this fragmentation.

A row overflows if it is updated and no longer fits in the data page where it was originally written. This usually happens as a result of an update of a VARCHAR or an ALTER TABLE statement.

overflow_creates - Overflow creates monitor element

The number of overflowed rows created on this table.

Table 840. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLE table function - Get table metrics	Always collected

Usage

package_id - Package identifier monitor element

A unique identifier for the package.

Table 841. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	Reported in the package list.	-

Usage

The value of this element matches a value from column PKGID of view SYSCAT.PACKAGES.

package_elapsed_time - Package elapsed time monitor element

The elapsed time spent executing sections within the package. Value is in milliseconds.

Table 842. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	Reported in the package list.	-

package_list_count - Package list count monitor element

The number of entries that are present within the package listing for a particular unit of work

Table 843. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-

package_list_exceeded - Package list exceeded monitor element

Indicates whether the number of packages used within the unit of work has exceeded the capacity of the package list. Possible values are YES and NO.

Table 844. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-

package_name - Package name monitor element

The name of the package that contains the SQL statement.

Table 845. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	Always collected

Table 846. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement
DCS Statement	dcs_stmt	Statement

Table 847. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details[1]	event_detailed_dlconn	-
Statements	event_stmt	-
Activities	event_activitystmt	-
Package cache	-	COLLECT BASE DATA

1 This option has been deprecated. Its use is no longer recommended and

might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

You may use this element to help identify the application program and the SQL statement that is executing.

package_schema - Package schema monitor element

The schema name of the package associated with an SQL statement.

Table 848. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	Always collected

Table 849. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Package cache	-	COLLECT BASE DATA

package_version_id - Package version monitor element

For a given package name and creator, there can exist (starting in DB2 Version 8) multiple versions. The package version identifies the version identifier of the package that contains the SQL statement currently executing. The version of a package is determined at precompile (PREP) of the embedded SQL program using the VERSION keyword. If not specified at precompile time the package version has a value of "" (empty string).

Table 850. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	Always collected

Table 851. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement

Table 852. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Statements	event_stmt	-
Activities	event_activitystmt	-
Package cache	-	COLLECT BASE DATA

Usage

Use this element to help identify the package and the SQL statement that is currently executing.

packet_receive_errors - Packet receive errors monitor element

Number of errors receiving packets since the network adapter started.

Table 853. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_NETWORK_RESOURCES table function - Return network adapter information	Always collected

packets_received - Packets received monitor element

Number of packets received since the network adapter started.

Table 854. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_NETWORK_RESOURCES table function - Return network adapter information	Always collected

packet_send_errors - Packet send errors monitor element

Number of errors sending packets since the network adapter started.

Table 855. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_NETWORK_RESOURCES table function - Return network adapter information	Always collected

packets_sent - Packets sent monitor element

Number of packets sent since the network adapter started.

Table 856. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_NETWORK_RESOURCES table function - Return network adapter information	Always collected

page_allocations - Page allocations monitor element

Number of pages that have been allocated to the index.

Table 857. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

page_reorgs - Page reorganizations monitor element

The number of page reorganizations executed for a table.

Table 858. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLE table function - Get table metrics	Always collected

Table 859. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table	Basic

For snapshot monitoring, this counter can be reset.

Table 860. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Tables	event_table	-

Usage

Although a page might have enough space, the page could become fragmented in the following situations:

- When a new row is inserted
- When an existing row is updated, and the update results in an increased record size

A page might require reorganization when it becomes fragmented. Reorganization moves all fragmented space to a contiguous area, where the new record can be

written. Such a page reorganization (page reorg) might require thousands of instructions. It also generates a log record of the operation.

Too many page reorganizations can result in less than optimal insert performance. You can use the REORG TABLE utility to reorganize a table and eliminate fragmentation. You can also use the APPEND parameter for the ALTER TABLE statement to indicate that all inserts are appended at the end of a table to avoid page reorganizations.

In situations where updates to rows causes the row length to increase, the page may have enough space to accommodate the new row, but a page reorg may be required to defragment that space. If the page does not have enough space for the new larger row, an overflow record is created causing *overflow_accesses* during reads. You can avoid both situations by using fixed length columns instead of varying length columns.

pages_from_block_ios - Total number of pages read by block I/O monitor element

The total number of pages read by block I/O into the block area of the buffer pool.

Table 861. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 862. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool	Buffer Pool

Usage

If block-based buffer pool is enabled, this element reports the total number of pages read by block I/O. Otherwise, this element returns 0.

To calculate the average number of pages sequentially prefetched per block-based I/O, divide the value of the **pages_from_block_ios** monitor element by the value of the **block_ios** monitor element. If this value is much less than the BLOCKSIZE option you have defined for the block-based buffer pool in the CREATE BUFFERPOOL or ALTER BUFFERPOOL statement, then block-based I/O is not being used to its full advantage. One possible cause for this is a mismatch between the extent size for the table space being sequentially prefetched and the block size of the block-based buffer pool.

pages_from_vectored_ios - Total number of pages read by vectored I/O monitor element

The total number of pages read by vectored I/O into the page area of the buffer pool.

Table 863. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 864. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool	Buffer Pool

pages_merged - Pages merged monitor element

Number of index pages that have been merged.

Table 865. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

pages_read - Number of pages read monitor element

The number of pages (data, index, and XML) read in from the physical table space containers for regular and large table spaces.

Table 866. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE

Usage

pages_written - Number of pages written monitor element

The number of pages (data, index, and XML) physically written to the table space container.

Table 867. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE

Usage

parent_activity_id - Parent activity ID monitor element

The unique ID of the activity's parent activity within the parent activity's unit of work. If there is no parent activity, the value of this monitor element is 0.

Table 868. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 869. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

Usage

Use this element along with the **parent_uow_id** element and **appl_id** element to uniquely identify the parent activity of the activity described in this activity record.

parent_uow_id - Parent unit of work ID monitor element

The unique unit of work identifier within an application handle. The ID of the unit of work in which the activity's parent activity originates. If there is no parent activity, the value is 0.

Table 870. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 871. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

Usage

Use this element along with the **parent_activity_id** element and **appl_id** element to uniquely identify the parent activity of the activity described in this activity record.

partial_record - Partial Record monitor element

Indicates that an event monitor record is only a partial record.

Table 872. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tables	event_table	-
Tablespaces	event_tablespace	-
Bufferpools	event_bufferpool	-
Connection	event_conn	-
Statements	event_stmt	-
Statements	event_subsection	-
Transactions	event_xact	-
Activities	event_activity	-

Usage

Most event monitors do not output their results until database deactivation. You can use the FLUSH EVENT MONITOR <monitorName> statement to force monitor values to the event monitor output writer. This allows you to force event monitor records to the writer without needing to stop and restart the event monitor. This element indicates whether an event monitor record was the result of flush operation and so is a partial record.

Flushing an event monitor does not cause its values to be reset. This means that a complete event monitor record is still generated when the event monitor is triggered.

At the event_activity logical data grouping, the possible values of **partial_record** monitor element are:

0 The activity record was generated normally at the end of activity.

1 The activity record was generated as a result of calling the WLM_CAPTURE_ACTIVITY_IN_PROGRESS stored procedure.

2 Information is missing for this activity because not enough storage was available to create the records. Information may be missing from the event_activity, event_activitystmt, or event_activityvals records.

participant_no - Participant within Deadlock

A sequence number uniquely identifying this participant within this deadlock.

Element identifier
participant_no

Element type
information

Table 873. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	event_dlconn	-

Table 873. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks with Details	event_detailed_dlconn	-

Usage Use this in your monitoring application to correlate deadlock connection event records with deadlock event records.

participant_no_holding_lk - Participant Holding a Lock on the Object Required by Application

The participant number of the application that is holding a lock on the object that this application is waiting to obtain.

Element identifier
participant_no_holding_lk

Element type
information

Table 874. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	event_dlconn	-
Deadlocks with Details	event_detailed_dlconn	-

Usage This element can help you determine which applications are in contention for resources.

partition_number - Partition Number

This element is only used in the target SQL tables by write-to-table event monitors in a partitioned database environment. This value indicates the number of the partition where event monitor data is inserted.

Table 875. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
-	-	-

passthru_time - Pass-Through Time

This element contains the aggregate amount of time, in milliseconds, that it has taken this data source to respond to PASSTHRU statements from all applications or a single application running on this federated server instance since the start of the federated server instance or the last reset of the database monitor counters, whichever is latest. The response time is measured as the difference between the time the federated server submits a PASSTHRU statement to the data source, and the time it takes the data source to respond, indicating that the statement has been processed.

Table 876. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Timestamp
Application	appl_remote	Timestamp

For snapshot monitoring, this counter can be reset.

Usage

Use this element to determine how much actual time is spent at this data source processing statements in pass-through mode.

passthrus - Pass-Through

This element contains a count of the total number of SQL statements that the federated server has passed through directly to this data source on behalf of any application since the start of the federated server instance or the last reset of the database monitor counters, whichever is latest.

Table 877. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

Usage Use this element to determine what percentage of your SQL statements can be handled natively by the federated server, and what percentage requires pass-through mode. If this value is high, you should determine the cause and investigate ways to better utilize native support.

piped_sorts_accepted - Piped Sorts Accepted

The number of piped sorts that have been accepted.

Table 878. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

For snapshot monitoring, this counter can be reset.

Usage Each active sort on the system allocates memory, which may result in sorting taking up too much of the available system memory.

When the number of accepted piped sorts is low compared to the number requested, you can improve sort performance by adjusting one or both of the following configuration parameters:

- sortheap
- sheapthres

If piped sorts are being rejected, you might consider decreasing your sort heap or increasing your sort heap threshold. You should be aware of the possible implications of either of these options. If you increase the sort heap threshold, then there is the possibility that more memory will remain allocated for sorting. This could cause the paging of memory to disk. If you decrease the sort heap, you might require an extra merge phase that could slow down the sort.

piped_sorts_requested - Piped Sorts Requested

The number of piped sorts that have been requested.

Table 879. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

For snapshot monitoring, this counter can be reset.

Usage Each active sort on the system allocates memory, which may result in sorting taking up too much of the available system memory.

The sort list heap (*sortheap*) and sort heap threshold (*sheapthres*) configuration parameters help to control the amount of memory used for sort operations. These parameters are also used to determine whether a sort will be piped.

Since piped sorts may reduce disk I/O, allowing more piped sorts can improve the performance of sort operations and possibly the performance of the overall system. A piped sort is not be accepted if the sort heap threshold will be exceeded when the sort heap is allocated for the sort. See *piped_sorts_accepted* for more information if you are experiencing piped sort rejections.

The SQL EXPLAIN output will show whether the optimizer requests a piped sort.

pkg_cache_inserts - Package cache inserts monitor element

The total number of times that a requested section was not available for use and had to be loaded into the package cache. This count includes any implicit prepares performed by the system.

Table 880. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE

Table 880. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 881. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 882. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

In conjunction with the **pkg_cache_lookups** monitor element, use this monitor element to calculate the package cache hit ratio using the following formula:

```
1 - (Package Cache Inserts / Package Cache Lookups)
```

pkg_cache_lookups - Package cache lookups monitor element

The number of times that an application looked for a section or package in the package cache. At a database level, it indicates the overall number of references since the database was started, or monitor data was reset. This counter includes the cases where the section is already loaded in the cache and when the section has to be loaded into the cache. In a concentrator environment where agents are being associated with different applications, additional package cache lookups may be required as a result of a new agent not having the required section or package available in local storage.

Table 883. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 884. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 885. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

To calculate the package cache hit ratio use the following formula:

```
1 - (Package Cache Inserts / Package Cache Lookups)
```

The package cache hit ratio tells you whether or not the package cache is being used effectively. If the hit ratio is high (more than 0.8), the cache is performing well. A smaller hit ratio may indicate that the package cache should be increased.

You will need to experiment with the size of the package cache to find the optimal number for the **pckcachesz** configuration parameter. For example, you might be able to use a smaller package cache size if there is no increase in the **pkg_cache_inserts** element when you decrease the size of the cache. Decreasing the package cache size frees up system resources for other work. It is also possible that you could improve overall system performance by increasing the size of the package cache if by doing so, you decrease the number of **pkg_cache_inserts**. This experimentation is best done under full workload conditions.

You can use this element with **ddl_sql_stmts** to determine whether or not the execution of DDL statements is impacting the performance of the package cache. Sections for dynamic SQL statements can become invalid when DDL statements are executed. Invalid sections are implicitly prepared by the system when next used. The execution of a DDL statement could invalidate a number of sections and the resulting extra overhead incurred when preparing those sections could significantly impact performance. In this case, the package cache hit ratio reflects the implicit recompilation of invalid sections. It does not reflect the insertion of new sections into the cache, so increasing the size of the package cache will not improve overall performance. You might find it less confusing to tune the cache for an application on its own before working in the full environment.

It is necessary to determine the role that DDL statements are playing in the value of the package cache hit ratio before deciding on what action to take. If DDL statements rarely occur, then cache performance may be improved by increasing its size. If DDL statements are frequent, then improvements may require that you limit the use of DDL statements (possibly to specific time periods).

The **static_sql_stmts** and **dynamic_sql_stmts** counts can be used to help provide information on the quantity and type of sections being cached.

Note: You may want to use this information at the database level to calculate the average package cache hit ratio all each applications. You should look at this information at an application level to find out the exact package cache hit ratio for a given application. It may not be worthwhile to increase the size of the package cache in order to satisfy the cache requirements of an application that only executes infrequently.

pkg_cache_num_overflows - Package Cache Overflows

The number of times that the package cache overflowed the bounds of its allocated memory.

Table 886. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

For snapshot monitoring, this counter can be reset.

Table 887. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage

Use this element with the **pkg_cache_size_top** monitor element to determine whether the size of the package cache needs to be increased to avoid overflowing.

pkg_cache_size_top - Package cache high watermark

The largest size reached by the package cache.

Note: The **pkg_cache_size_top** monitor element is deprecated starting with DB2 Version 9.5. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

Table 888. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Table 889. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage

If the package cache overflowed, then this element contains the largest size reached by the package cache during the overflow.

Check the **pkg_cache_num_overflows** monitor element to determine if such a condition occurred.

You can determine the minimum size of the package cache required by your workload by:

```
maximum package cache size / 4096
```

Rounding the result up to a whole number, indicates the minimum number of 4K pages required by the package cache to avoid overflow.

pool_async_data_read_reqs - Buffer pool asynchronous read requests monitor element

The number of asynchronous read requests by the prefetcher to the operating system. These requests are typically large block I/Os of multiple pages.

Table 890. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 891. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 892. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-

Usage

To calculate the average number of data pages in each read request, use the following formula:

```
pool_async_data_reads / pool_async_data_read_reqs
```

This average can help you determine the average read I/O size used by the prefetcher. This data can also be helpful in understanding the large block I/O requirements of the measured workload.

The maximum size of a prefetcher read I/O is the value specified on the EXTENTSIZE option of the CREATE TABLESPACE statement for the table space involved, but it can be smaller under some circumstances:

- when some pages of the extent are already in the buffer pool
- when exceeding operating system capabilities
- when the EXTENTSIZE option value is very large, such that doing a large I/O would be detrimental to overall performance

pool_async_data_reads - Buffer pool asynchronous data reads monitor element

Indicates the number of data pages read in from the table space containers (physical) by asynchronous engine dispatchable units (EDUs) for all types of table spaces.

Table 893. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 894. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 895. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-

Usage

You can use this element with **pool_data_p_reads** to calculate the number of physical reads that were performed synchronously (that is, physical data page reads that were performed by database manager agents). Use the following formula:

```
1-((pool_data_p_reads+pool_index_p_reads)-(pool_async_data_reads+pool_async_index_reads))
/(pool_data_l_reads+pool_index_l_reads)
```

By comparing the ratio of asynchronous to synchronous reads, you can gain insight into how well the prefetchers are working. This element can be helpful when you are tuning the **num_ioservers** configuration parameter.

Asynchronous reads are performed by database manager prefetchers.

pool_async_data_writes - Buffer pool asynchronous data writes monitor element

The number of times a buffer pool data page was physically written to disk by either an asynchronous page cleaner, or a prefetcher. A prefetcher may have written dirty pages to disk to make space for the pages being prefetched.

Table 896. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 897. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 898. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-

Usage You can use this element with the **pool_data_writes** monitor element to calculate the number of physical write requests that were performed synchronously (that is, physical data page writes that were performed by database manager agents). Use the following formula:

```
pool_data_writes - pool_async_data_writes
```

By comparing the ratio of asynchronous to synchronous writes, you can gain insight into how well the buffer pool page cleaners are performing. This ratio can be helpful when you are tuning the **num_iocleaners** configuration parameter.

pool_async_index_read_reqs - Buffer pool asynchronous index read requests monitor element

The number of asynchronous read requests for index pages.

Table 899. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE

Table 899. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 900. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 901. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-

Usage To calculate the number of index pages read per asynchronous request, use the following formula:

```
pool_async_index_reads / pool_async_index_read_reqs
```

This average can help you determine the amount of asynchronous I/O done for index pages in each interaction with the prefetcher.

pool_async_index_reads - Buffer pool asynchronous index reads monitor element

Indicates the number of index pages read in from the table space containers (physical) by asynchronous engine dispatchable units (EDUs) for all types of table spaces.

Table 902. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 903. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 904. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-

Usage

You can use this element with the **pool_index_p_reads** monitor element to calculate the number of physical reads that were performed synchronously (that is, physical index page reads that were performed by database manager agents). Use the following formula:

```
1 - ((pool_data_p_reads + pool_index_p_reads) - (pool_async_data_reads + pool_async_index_reads))
 / (pool_data_l_reads + pool_index_l_reads)
```

By comparing the ratio of asynchronous to synchronous reads, you can gain insight into how well the prefetchers are working. This element can be helpful when you are tuning the **num_ioservers** configuration parameter.

Asynchronous reads are performed by database manager prefetchers.

pool_async_index_writes - Buffer pool asynchronous index writes monitor element

The number of times a buffer pool index page was physically written to disk by either an asynchronous page cleaner, or a prefetcher. A prefetcher may have written dirty pages to disk to make space for the pages being prefetched.

Table 905. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 906. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 907. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-

Usage

You can use this element with the **pool_index_writes** monitor element to calculate the number of physical index write requests that were performed synchronously (that is, physical index page writes that were performed by database manager agents). Use the following formula:

```
pool_index_writes - pool_async_index_writes
```

By comparing the ratio of asynchronous to synchronous writes, you can gain insight into how well the buffer pool page cleaners are performing. This ratio can be helpful when you are tuning the **num_iocleaners** configuration parameter.

pool_async_read_time - Buffer Pool Asynchronous Read Time

Indicates the total amount of time spent reading in data and index pages from the table space containers (physical) by asynchronous engine dispatchable units (EDUs) for all types of table spaces. This value is given in milliseconds.

Element identifier
pool_async_read_time

Element type
counter

Table 908. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 909. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-

Usage You can use this element to calculate the elapsed time for synchronous reading, using the following formula:

```
pool_read_time - pool_async_read_time
```

You can also use this element to calculate the average asynchronous read time using the following formula:

```
pool_async_read_time / pool_async_data_reads
```

These calculations can be used to understand the I/O work being performed.

pool_async_write_time - Buffer pool asynchronous write time monitor element

The total elapsed time spent writing data or index pages from the buffer pool to disk by database manager page cleaners. This value is reported in milliseconds.

Table 910. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 911. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-

Usage

To calculate the elapsed time spent writing pages synchronously, use the following formula:

```
pool_write_time - pool_async_write_time
```

You can also use this element to calculate the average asynchronous write time using the following formula:

```
pool_async_write_time
 / (pool_async_data_writes
   + pool_async_index_writes)
```

These calculations can be used to understand the I/O work being performed.

pool_async_xda_read_reqs - Buffer pool asynchronous XDA read requests monitor element

The number of asynchronous read requests for XML storage object (XDA) data.

Table 912. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 913. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool

Table 913. Snapshot Monitoring Information (continued)

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 914. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-

Usage To calculate the average number of XML storage object data pages read per asynchronous request, use the following formula:

```
pool_async_xda_reads / pool_async_xda_read_reqs
```

This average can help you determine the amount of asynchronous I/O done in each interaction with the prefetcher.

pool_async_xda_reads - Buffer pool asynchronous XDA data reads monitor element

Indicates the number of XML storage object (XDA) data pages read in from the table space containers (physical) by asynchronous engine dispatchable units (EDUs) for all types of table spaces.

Table 915. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 916. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 917. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-

Usage

Use the **pool_async_xda_reads** and **pool_xda_p_reads** monitor elements to calculate the number of physical reads that were performed synchronously on XML storage object data pages (that is, physical data page reads that were performed by database manager agents on XML data). Use the following formula:

```
pool_xda_p_reads - pool_async_xda_reads
```

By comparing the ratio of asynchronous to synchronous reads, you can gain insight into how well the prefetchers are working. This element can be helpful when you are tuning the **num_ioservers** configuration parameter.

Asynchronous reads are performed by database manager prefetchers.

pool_async_xda_writes - Buffer pool asynchronous XDA data writes monitor element

The number of times a buffer pool data page for an XML storage object (XDA) was physically written to disk by either an asynchronous page cleaner, or a prefetcher. A prefetcher may have written dirty pages to disk to make space for the pages being prefetched.

Table 918. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 919. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 920. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-

Usage You can use this element with the **pool_xda_writes** monitor element to calculate the number of physical write requests that were performed synchronously on XML storage object data pages (that is, physical data page writes that were performed by database manager agents on XML data). Use the following formula:

```
pool_xda_writes - pool_async_xda_writes
```

By comparing the ratio of asynchronous to synchronous writes, you can gain insight into how well the buffer pool page cleaners are performing. This ratio can be helpful when you are tuning the **num_iocleaners** configuration parameter.

pool_config_size - Configured Size of Memory Pool

The internally configured size of a memory pool in DB2 database system. The value is given in bytes.

Table 921. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	memory_pool	Basic
Database	memory_pool	Basic
Application	memory_pool	Basic

Table 922. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_dbmemuse	-
Connection	event_connmemuse	-

Usage To track system memory usage, use this value in conjunction with **pool_cur_size**, **pool_id**, and **pool_watermark**.

To see if a memory pool is nearly full, compare **pool_config_size** to **pool_cur_size**. For example, assume that the utility heap is too small. You can diagnose this specific problem by taking snapshots at regular intervals, and looking in the utility heap section of the snapshot output. If required, the **pool_cur_size** might be allowed to exceed the **pool_config_size** to prevent an out of memory failure. If this occurs very infrequently, no further action is likely required. However if **pool_cur_size** is consistently close to or larger than **pool_config_size**, you might consider increasing the size of the utility heap.

pool_cur_size - Current Size of Memory Pool

The current size of a memory pool. The value is given in bytes.

Table 923. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	memory_pool	Basic
Database	memory_pool	Basic
Application	memory_pool	Basic

Table 924. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_dbmemuse	-
Connection	event_connmemuse	-

Usage To track system memory usage, use this value in conjunction with *pool_config_size*, *pool_id*, and *pool_watermark*.

To see if a memory pool is nearly full, compare *pool_config_size* to *pool_cur_size*. For example, assume that the utility heap is too small. You can diagnose this specific problem by taking snapshots at regular intervals, and looking in the utility heap section of the snapshot output. If the value of *pool_cur_size* is consistently close to *pool_config_size*, you may want to consider increasing the size of the utility heap.

pool_data_l_reads - Buffer pool data logical reads monitor element

The number of data pages which have been requested from the buffer pool (logical) for regular and large table spaces.

Table 925. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 926. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 927. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

This count includes accesses to data that is:

- Already in the buffer pool when the database manager needs to process the page.
- Read into the buffer pool before the database manager can process the page.

Use the **pool_data_l_reads** and **pool_data_p_reads** monitor elements to calculate the overall data page hit ratio for the buffer pool using the following formula:

```
1 - ((pool_data_p_reads - pool_async_data_reads) / pool_data_l_reads)
```

Increasing buffer pool size will generally improve the hit ratio, but you will reach a point of diminishing return. Ideally, if you could allocate a buffer pool large enough to store your entire database, then once the system is up and running you would get a hit ratio of 100%. However, this is unrealistic in most cases. The significance of the hit ratio really depends on the size of your data, and the way it is accessed. A very large database where data is accessed evenly would have a poor hit ratio. There is little you can do with very large tables.

To improve hit ratios for smaller, frequently accessed tables and indexes, assign them to individual buffer pools.

pool_data_p_reads - Buffer pool data physical reads monitor element

Indicates the number of data pages read in from the table space containers (physical) for regular and large table spaces.

Table 928. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 929. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 930. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element with the **pool_data_l_reads** and **pool_async_data_reads** monitor elements to calculate the number of physical reads that were performed synchronously (that is, physical data page reads that were performed by database manager agents). Use the following formula:

```
1 - ((pool_data_p_reads + pool_index_p_reads) - (pool_async_data_reads + pool_async_index_reads))
 / (pool_data_l_reads + pool_index_l_reads)
```

By comparing the ratio of asynchronous to synchronous reads, you can gain insight into how well the prefetchers are working. This information can be helpful when you are tuning the **num_ioservers** configuration parameter.

pool_data_writes - Buffer pool data writes monitor element

The number of times a buffer pool data page was physically written to disk.

Table 931. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 932. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 933. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

If a buffer pool data page is written to disk for a high percentage of the value of the **pool_data_p_reads** monitor element, you may be able to improve performance by increasing the number of buffer pool pages available for the database.

A buffer pool data page is written to disk for the following reasons:

- To free a page in the buffer pool so another page can be read
- To flush the buffer pool

The system does not always write a page to make room for a new one. If the page has not been updated, it can simply be replaced. This replacement is not counted for this element.

The data page can be written by an asynchronous page-cleaner agent before the buffer pool space is required, as reported by the **pool_async_data_writes** monitor element. These asynchronous page writes are included in the value of this element in addition to synchronous page writes.

When calculating this percentage, disregard the number of physical reads required to initially fill the buffer pool. To determine the number of pages written:

1. Run your application (to load the buffer).
2. Note the value of this element.
3. Run your application again.
4. Subtract the value recorded in step 2 from the new value of this element.

In order to prevent the buffer pool from being deallocated between the runnings of your application, you should do one of the following:

- Activate the database with the **ACTIVATE DATABASE** command.
- Have an idle application connected to the database.

If all applications are updating the database, increasing the size of the buffer pool may not have much impact on performance since most of the buffer pool pages contain updated data, which must be written to disk. However, if the updated pages can be used by other units of work before being written out, the buffer pool can save a write and a read, which will improve your performance.

pool_drty_pg_steal_clns - Buffer pool victim page cleaners triggered monitor element

The number of times a page cleaner was invoked because a synchronous write was needed during the victim buffer replacement for the database.

Table 934. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE

Table 935. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 936. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage

Using the following formula, you may calculate what percentage of all cleaner invocations are represented by this element:

```
   pool_drty_pg_steal_clns
/ (pool_drty_pg_steal_clns
 + pool_drty_pg_thrsh_clns
 + pool_lsn_gap_clns)
```

If this ratio is low, it may indicate that you have defined too many page cleaners. If your **chngpgs_thresh** configuration parameter is set too low, you may be writing out pages that you will dirty later. Aggressive cleaning defeats one purpose of the buffer pool, that is to defer writing to the last possible moment.

If this ratio is high, it may indicate that you have not defined enough page cleaners. Not having enough page cleaners increases recovery time after failures.

When the DB2_USE_ALTERNATE_PAGE_CLEANING registry variable is OFF:

- The **pool_drty_pg_steal_clns** monitor element is inserted into the monitor stream.

- The **pool_drty_pg_steal_clns** monitor element counts the number of times a page cleaner was invoked because a synchronous write was needed during the victim buffer replacement for the database.

When the DB2_USE_ALTERNATE_PAGE_CLEANING registry variable is ON:

- The **pool_drty_pg_steal_clns** monitor element inserts 0 into the monitor stream.
- There is no explicit triggering of the page cleaners when a synchronous write is needed during victim buffer replacement. To determine whether or not the right number of page cleaners is configured for the database or for specific buffer pools, please refer to the **pool_no_victim_buffer** monitor element.

Note: Although dirty pages are written out to disk, the pages are not removed from the buffer pool right away, unless the space is needed to read in new pages.

pool_drty_pg_thrsh_clns - Buffer pool threshold cleaners triggered monitor element

The number of times a page cleaner was invoked because a buffer pool had reached the dirty page threshold criterion for the database.

Table 937. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE

Table 938. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 939. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage The threshold is set by the **chngpgs_thresh** configuration parameter. It is a percentage applied to the buffer pool size. When the number of dirty pages in the pool exceeds this value, the cleaners are triggered.

If the **chngpgs_thresh** configuration parameter value is set too low, pages might be written out too early, requiring them to be read back in. If it is set too high, then too many pages may accumulate, requiring users to write out pages synchronously.

When the DB2_USE_ALTERNATE_PAGE_CLEANING registry variable is OFF:

- The **pool_drty_pg_thrsh_clns** monitor element is inserted into the monitor stream.
- The **pool_drty_pg_thrsh_clns** monitor element counts the number of times a page cleaner was invoked because a buffer pool had reached the dirty page threshold criterion for the database.

When the DB2_USE_ALTERNATE_PAGE_CLEANING registry variable is ON:

- The **pool_drty_pg_thrsh_clns** monitor element inserts 0 into the monitor stream.
- Page cleaners are always active, attempting to ensure there are sufficient free buffers for victims available instead of waiting to be triggered by the criterion value.

pool_id - Memory Pool Identifier

The type of memory pool.

Table 940. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	memory_pool	Basic
Database	memory_pool	Basic
Application	memory_pool	Basic

Table 941. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_dbmemuse	-
Connection	event_connmemuse	-

Usage

To track system memory usage, use this value in conjunction with **pool_config_size**, **pool_cur_size**, and **pool_watermark**.

Use **pool_id** to identify the memory pools discussed in the system monitor output. The various memory pool identifiers can be found in `sqlmon.h`. Under normal operating conditions, one or more of each of the following pools can be expected.

API Constant	Description
SQLM_HEAP_APPLICATION	Application Heap
SQLM_HEAP_DATABASE	Database Heap
SQLM_HEAP_LOCK_MGR	Lock Manager Heap
SQLM_HEAP_UTILITY	Backup/Restore/Utility Heap
SQLM_HEAP_STATISTICS	Statistics Heap
SQLM_HEAP_PACKAGE_CACHE	Package Cache Heap
SQLM_HEAP_CAT_CACHE	Catalog Cache Heap
SQLM_HEAP_MONITOR	Database Monitor Heap
SQLM_HEAP_STATEMENT	Statement Heap
SQLM_HEAP_FCMBP	FCMBP Heap
SQLM_HEAP_IMPORT_POOL	Import Pool
SQLM_HEAP_OTHER	Other Memory
SQLM_HEAP_BP	Buffer Pool Heap
SQLM_HEAP_APPL_SHARED	Applications Shared Heap

API Constant	Description
SQLM_HEAP_SHARED_SORT	Sort Shared Heap

pool_index_l_reads - Buffer pool index logical reads monitor element

Indicates the number of index pages which have been requested from the buffer pool (logical) for regular and large table spaces.

Table 942. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 943. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool

Table 943. Snapshot Monitoring Information (continued)

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 944. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

This count includes accesses to index pages that are:

- Already in the buffer pool when the database manager needs to process the page
- Read into the buffer pool before the database manager can process the page.

In conjunction with **pool_index_p_reads**, you can calculate the index page hit ratio for the buffer pool using the following formula:

```
1 - ((pool_index_p_reads - pool_async_index_reads) / pool_index_l_reads)
```

Use the **pool_data_l_reads** and **pool_data_p_reads** monitor elements to calculate the overall data page hit ratio for the buffer pool using the following formula:

```
1 - ((pool_data_p_reads - pool_async_data_reads) / pool_data_l_reads)
```

If the hit ratio is low, increasing the number of buffer pool pages may improve performance.

pool_index_p_reads - Buffer pool index physical reads monitor element

Indicates the number of index pages read in from the table space containers (physical) for regular and large table spaces.

Table 945. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 946. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 947. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

In conjunction with the **pool_index_l_reads** monitor element, you can calculate the index page hit ratio for the buffer pool using the following formula:

```
1 - ((pool_index_p_reads - pool_async_index_reads) / pool_index_l_reads)
```

pool_index_writes - Buffer pool index writes monitor element

Indicates the number of times a buffer pool index page was physically written to disk.

Table 948. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 949. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 950. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Like a data page, a buffer pool index page is written to disk for the following reasons:

- To free a page in the buffer pool so another page can be read
- To flush the buffer pool

The system does not always write a page to make room for a new one. If the page has not been updated, it can simply be replaced. This replacement is not counted for this element.

The index page can be written by an asynchronous page-cleaner agent before the buffer pool space is required. These asynchronous index page writes are included in the value of this element in addition to synchronous index page writes (see the **pool_async_index_writes** monitor element).

If a buffer pool index page is written to disk for a high percentage of the value of the **pool_index_p_reads** monitor element, you may be able to improve performance by increasing the number of buffer pool pages available for the database.

When calculating this percentage, disregard the number of physical reads required to initially fill the buffer pool. To determine the number of pages written:

1. Run your application (to load the buffer).
2. Note the value of this element.
3. Run your application again.
4. Subtract the value recorded in step 2 from the new value of this element.

In order to prevent the buffer pool from being deallocated between the runnings of your application, you should do one of the following:

- Activate the database with the **ACTIVATE DATABASE** command.
- Have an idle application connected to the database.

If all applications are updating the database, increasing the size of the buffer pool may not have much impact on performance, since most of the pages contain updated data which must be written to disk.

pool_lsn_gap_clns - Buffer pool log space cleaners triggered monitor element

The number of times a page cleaner was invoked because the logging space used had reached a predefined criterion for the database.

Table 951. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE

Table 952. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 953. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage

This element can be used to help evaluate whether you have enough space for logging, and whether you need more log files or larger log files.

The page cleaning criterion is determined by the setting for the **softmax** configuration parameter. Page cleaners are triggered if the oldest page in the buffer pool contains an update described by a log record that is older than the current log position by the criterion value.

When the DB2_USE_ALTERNATE_PAGE_CLEANING registry variable is OFF:

- The **pool_lsn_gap_clns** monitor element is inserted into the monitor stream.
- Page cleaners are triggered if the oldest page in the buffer pool contains an update described by a log record that is older than the current log position by the criterion value.

When the DB2_USE_ALTERNATE_PAGE_CLEANING registry variable is ON:

- The **pool_lsn_gap_clns** monitor element inserts 0 into the monitor stream.
- Page cleaners write pages proactively instead of waiting to be triggered by the criterion value.

pool_no_victim_buffer - Buffer pool no victim buffers monitor element

Number of times an agent did not have a preselected victim buffer available.

Table 954. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE

Table 955. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Tablespace	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 956. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespace	event_tablespace	-

Usage This element can be used to help evaluate whether you have enough page cleaners for a given buffer pool when using proactive page cleaning.

When the DB2_USE_ALTERNATE_PAGE_CLEANING registry variable is ON, the pool_no_victim_buffer element counts the number of times that an agent did not find a preselected victim buffer available for immediate use, and was forced to search the buffer pool for a suitable victim buffer.

If the value of pool_no_victim_buffer element is high relative to the number of logical reads in the buffer pool, then the DB2 database system is having difficulty ensuring that sufficient numbers of good victims are available for use. Increasing the number of page cleaners will increase the ability of DB2 to provide preselected victim buffers.

When the DB2_USE_ALTERNATE_PAGE_CLEANING registry variable is OFF, the pool_no_victim_buffer element has no predictive value, and can be safely ignored. In this configuration, the DB2 database system does not attempt to ensure that agents have preselected victim buffers available to them, so most accesses to the buffer pool will require that the agent search the buffer pool to find a victim buffer.

pool_read_time - Total buffer pool physical read time monitor element

Indicates the total amount of time spent reading in data and index pages from the table space containers (physical) for all types of table spaces. This value is given in milliseconds.

Table 957. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 958. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

Table 958. Snapshot Monitoring Information (continued)

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 959. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

You can use this element with **pool_data_p_reads** and **pool_index_p_reads** monitor elements to calculate the average page-read time. This average is important since it may indicate the presence of an I/O wait, which in turn may indicate that you should be moving data to a different device.

At the database and table space levels, this element includes the value of the **pool_async_read_time** monitor element.

pool_secondary_id - Memory Pool Secondary Identifier

An additional identifier to help determine the memory pool for which monitor data is returned.

Element identifier
pool_secondary_id

Element type
Information

Table 960. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	memory_pool	Basic

Table 960. Snapshot Monitoring Information (continued)

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	memory_pool	Basic
Application	memory_pool	Basic

Table 961. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_dbmemuse	-
Connection	event_connmemuse	-

Usage Use together with pool_id to determine the memory pool for which monitor data is returned. Data for pool_secondary_id only appears when necessary. For example, it appears when the pool_id indicated is Buffer Pool Heap to determine which buffer pool the monitor data relates to.

When a database is created, it has a default buffer pool, called IBMDEFAULTBP, with a size determined by the platform. This buffer pool has a secondary id of "1". In addition to this buffer pool and any buffer pools that you create, a set of system buffer pools are created by default, each corresponding to a different page size. IDs for these buffer pools can appear in snapshots for pool_secondary_id:

- System 32k buffer pool
- System 16k buffer pool
- System 8k buffer pool
- System 4k buffer pool

pool_temp_data_l_reads - Buffer pool temporary data logical reads monitor element

Indicates the number of data pages which have been requested from the buffer pool (logical) for temporary table spaces.

Table 962. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE

Table 962. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 963. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 964. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-

Table 964. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

In conjunction with the **pool_temp_data_p_reads** element, a calculation for the data page hit ratio for buffer pools located in temporary table spaces can be made using the following formula:

```
1 - (pool_temp_data_p_reads / pool_temp_data_l_reads)
```

The overall buffer pool hit ratio can be calculated as follows:

```
    1 - ((pool_data_p_reads + pool_xda_p_reads +
pool_index_p_reads + pool_temp_data_p_reads
+ pool_temp_xda_p_reads + pool_temp_index_p_reads )
     / (pool_data_l_reads + pool_xda_l_reads + pool_index_l_reads +
     pool_temp_data_l_reads + pool_temp_xda_l_reads
     + pool_temp_index_l_reads )) * 100%
```

This calculation takes into account all of the pages (index and data) that are cached by the buffer pool.

pool_temp_data_p_reads - Buffer pool temporary data physical reads monitor element

Indicates the number of data pages read in from the table space containers (physical) for temporary table spaces.

Table 965. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 965. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 966. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 967. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Database	event_db	-

Table 967. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Tablespaces	event_tablespace	-
Connection	event_conn	-
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

The functionality to record buffer pool information at the statement level is supported for API and CLP snapshot requests.

In conjunction with the **pool_temp_data_l_reads** element, a calculation for the data page hit ratio for buffer pools located in temporary table spaces can be made using the following formula:

```
1 - (pool_temp_data_p_reads / pool_temp_data_l_reads)
```

pool_temp_index_l_reads - Buffer pool temporary index logical reads monitor element

Indicates the number of index pages which have been requested from the buffer pool (logical) for temporary table spaces.

Table 968. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE

Table 968. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 969. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 970. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-

Table 970. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element, in conjunction with the **pool_temp_index_p_reads** element, to calculate the index page hit ratio for buffer pools located in temporary table spaces, using the following formula:

```
1 - (pool_temp_index_p_reads / pool_temp_index_l_reads)
```

pool_temp_index_p_reads - Buffer pool temporary index physical reads monitor element

Indicates the number of index pages read in from the table space containers (physical) for temporary table spaces.

Table 971. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 971. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 972. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 973. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE

Table 973. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element, in conjunction with the **pool_temp_index_l_reads** element, to calculate the index page hit ratio for buffer pools located in temporary table spaces, using the following formula:

```
1 - (pool_temp_index_p_reads / pool_temp_index_l_reads)
```

pool_temp_xda_l_reads - Buffer pool temporary XDA data logical reads monitor element

Indicates the number of pages for XML storage object (XDA) data which have been requested from the buffer pool (logical) for temporary table spaces.

Table 974. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 974. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 975. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 976. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

You can use the **pool_temp_xda_l_reads** monitor element in conjunction with **pool_temp_xda_p_reads**, **pool_temp_data_l_reads**, and **pool_temp_data_p_reads**

monitor elements to calculate the data page hit ratio for buffer pools located in temporary table spaces by using the following formula:

```
1 - ((pool_temp_data_p_reads + pool_temp_xda_p_reads)
      / (pool_temp_data_l_reads + pool_temp_xda_l_reads))
```

pool_temp_xda_p_reads - Buffer pool temporary XDA data physical reads monitor element

Indicates the number of pages for XML storage object (XDA) data read in from the table space containers (physical) for temporary table spaces.

Table 977. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 978. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 979. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

You can use the **pool_temp_xda_p_reads** monitor element in conjunction with **pool_temp_xda_l_reads**, **pool_temp_data_l_reads**, and **pool_temp_data_p_reads** monitor elements to calculate the data page hit ratio for buffer pools located in temporary table spaces by using the following formula:

```
1 - ((pool_temp_data_p_reads + pool_temp_xda_p_reads) /
(pool_temp_data_l_reads + pool_temp_xda_l_reads))
```

pool_watermark - Memory Pool Watermark

The largest size of a memory pool since its creation. The value is given in bytes.

Element identifier
pool_watermark

Element type
Information

Table 980. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	memory_pool	Basic
Database	memory_pool	Basic
Application	memory_pool	Basic

Table 981. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_dbmemuse	-
Connection	event_connmemuse	-

Usage On continuously running systems, you can use the *pool_watermark* and *pool_config_size* elements together to predict potential memory problems.

For example, take a snapshot at regular intervals (for instance, daily), and examine the *pool_watermark* and *pool_config_size* values. If you observe that the value of *pool_watermark* is becoming increasingly close to *pool_config_size* (a premature indication of potential future memory-related problems), this may indicate that you should increase the size of the memory pool.

pool_write_time - Total buffer pool physical write time monitor element

Provides the total amount of time spent physically writing data or index pages from the buffer pool to disk. Elapsed time is given in milliseconds.

Table 982. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 982. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 983. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 984. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE

Table 984. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element with **pool_data_writes** and **pool_index_writes** monitor elements to calculate the average page-write time. This average is important since it may indicate the presence of an I/O wait, which in turn may indicate that you should be moving data to a different device.

At the database and table space levels, this element includes the value of the **pool_async_write_time** monitor element.

pool_xda_l_reads - Buffer pool XDA data logical reads monitor element

Indicates the number of data pages for XML storage objects (XDAs) which have been requested from the buffer pool (logical) for regular and large table spaces.

Table 985. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE

Table 985. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 986. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 987. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	-
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

This count includes accesses to data that is:

- Already in the buffer pool when the database manager needs to process the page
- Read into the buffer pool before the database manager can process the page.

Use the **pool_xda_l_reads**, **pool_xda_p_reads**, **pool_data_l_reads**, and **pool_data_p_reads** monitor elements to calculate the data page hit ratio for the buffer pool by using the following formula:

```
1 - ((pool_data_p_reads + pool_xda_p_reads)
   / (pool_data_l_reads + pool_xda_l_reads))
```

The overall buffer pool hit ratio can be calculated as follows:

```
    1 - ((pool_data_p_reads + pool_xda_p_reads +
pool_index_p_reads + pool_temp_data_p_reads
+ pool_temp_xda_p_reads + pool_temp_index_p_reads )
     / (pool_data_l_reads + pool_xda_l_reads + pool_index_l_reads +
     pool_temp_data_l_reads + pool_temp_xda_l_reads
     + pool_temp_index_l_reads )) * 100%
```

This calculation takes into account all of the pages (index and data) that are cached by the buffer pool.

Increasing buffer pool size will generally improve the hit ratio, but you will reach a point of diminishing return. Ideally, if you could allocate a buffer pool large enough to store your entire database, then once the system is up and running you would get a hit ratio of 100%. However, this is unrealistic in most cases. The significance of the hit ratio depends on the size of your data, and the way it is accessed. A very large database where data is accessed evenly would have a poor hit ratio. There is little you can do with very large tables. In such case, you would focus your attention on smaller, frequently accessed tables, and on the indexes.

pool_xda_p_reads - Buffer pool XDA data physical reads monitor element

Indicates the number of data pages for XML storage objects (XDAs) read in from the table space containers (physical) for regular and large table spaces.

Table 988. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 988. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 989. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 990. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE

Table 990. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use the **pool_async_xda_reads** and **pool_xda_p_reads** monitor elements to calculate the number of physical reads that were performed synchronously on XML storage object data pages (that is, physical data page reads that were performed by database manager agents on XML data). Use the following formula:

```
pool_xda_p_reads - pool_async_xda_reads
```

By comparing the ratio of asynchronous to synchronous reads, you can gain insight into how well the prefetchers are working. This element can be helpful when you are tuning the **num_ioservers** configuration parameter.

Use the **pool_xda_l_reads**, **pool_xda_p_reads**, **pool_data_l_reads**, and **pool_data_p_reads** monitor elements to calculate the data page hit ratio for the buffer pool by using the following formula:

```
1 - ((pool_data_p_reads + pool_xda_p_reads)
   / (pool_data_l_reads + pool_xda_l_reads))
```

pool_xda_writes - Buffer pool XDA data writes monitor element

Indicates the number of times a buffer pool data page for an XML storage object (XDA) was physically written to disk.

Table 991. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 991. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 992. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 993. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE

Table 993. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

This monitor element helps you to assess whether performance may be improved by increasing the number of buffer pool pages available for the database. For databases containing XML data, you should consider the ratio of buffer pool page writes to buffer pool page reads both for XML data (using the **pool_xda_writes** and the **pool_xda_p_reads** monitor elements) and for relational data types (using the **pool_data_writes** and the **pool_data_p_reads** monitor elements).

Use the **pool_xda_l_reads**, **pool_xda_p_reads**, **pool_data_l_reads**, and **pool_data_p_reads** monitor elements to calculate the data page hit ratio for the buffer pool by using the following formula:

```
1 - ((pool_data_p_reads + pool_xda_p_reads)
   / (pool_data_l_reads + pool_xda_l_reads))
```

post_shrthreshold_hash_joins - Post threshold hash joins

The total number of hash joins that were throttled back by the sort memory throttling algorithm. A throttled hash join is a hash join that was granted less memory than requested by the sort memory manager.

Table 994. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	-

For snapshot monitoring, this counter can be reset.

Table 995. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

A hash join is throttled back when the memory allocation from the shared sort heap is close to the limit set by database configuration parameter *sheapthres_shr*. This throttling will significantly reduce the number of overflows over *sheapthres_shr*

limit in a system that is not properly configured. The data reported in this element only reflects hash joins using memory allocated from the shared sort heap.

post_shrthreshold_sorts - Post shared threshold sorts monitor element

The total number of sorts that were throttled back by the sort memory throttling algorithm. A throttled sort is a sort that was granted less memory than requested by the sort memory manager.

Table 996. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 997. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Sort

For snapshot monitoring, this counter can be reset.

Table 998. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Database	event_db	-
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

A sort is throttled back when the memory allocation for sorts is close to the limit set by database configuration parameter **sheapthres_shr**. This throttling will significantly reduce the number of overflows over **sheapthres_shr** limit in a system that is not properly configured. The data reported in this element only reflects sorts using memory allocated from the shared sort heap.

post_threshold_hash_joins - Hash Join Threshold

The total number of times that a hash join heap request was limited due to concurrent use of shared or private sort heap space.

Element identifier
post_threshold_hash_joins

Element type
counter

Table 999. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

For snapshot monitoring, this counter can be reset.

Usage If this value is large (greater than 5% of hash_join_overflows), the sort heap threshold should be increased.

post_threshold_olap_funcs - OLAP Function Threshold monitor element

The number of OLAP functions that have requested a sort heap after the sort heap threshold has been exceeded.

Table 1000. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

For snapshot monitoring, this counter can be reset.

Usage

Sorts, hash joins, and OLAP functions are examples of operations which utilize a sort heap. Under normal conditions, the database manager will allocate sort heap using the value specified by the sortheap configuration parameter. If the amount of memory allocated to sort heaps exceeds the sort heap threshold (sheapthres configuration parameter), the database manager will allocate subsequent sort heaps using a value less than that specified by the sortheap configuration parameter.

OLAP functions which start after the sort heap threshold has been reached may not receive an optimum amount of memory to execute.

To improve sort, hash join, OLAP function performance, and overall system performance, modify the sort heap threshold and sort heap size configuration parameters.

If this element's value is high, increase the sort heap threshold (sheapthres).

post_threshold_sorts - Post threshold sorts monitor element

The number of sorts that have requested heaps after the sort heap threshold has been exceeded.

Table 1001. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 1001. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 1002. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Sort

For snapshot monitoring, this counter can be reset.

Table 1003. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Under normal conditions, the database manager will allocate sort heap using the value specified by the **sortheap** configuration parameter. If the amount of memory allocated to sort heaps exceeds the sort heap threshold (**sheapthres** configuration parameter), the database manager will allocate sort heap using a value less than that specified by the **sortheap** configuration parameter.

Each active sort on the system allocates memory, which may result in sorting taking up too much of the system memory available. Sorts that start after the sort heap threshold has been reached may not receive an optimum amount of memory to execute, but, as a result, the entire system may benefit. By modifying the sort heap threshold and sort heap size configuration parameters, sort operation performance and overall system performance can be improved. If this element's value is high, you can:

- Increase the sort heap threshold (**sheapthres**) or,
- Adjust applications to use fewer or smaller sorts via SQL query changes.

prefetch_wait_time - Time waited for prefetch monitor element

The time an application spent waiting for an I/O server (prefetcher) to finish loading pages into the buffer pool. The value is given in milliseconds.

Table 1004. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Application	appl	Buffer Pool

Table 1005. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage This element can be used to experiment with changing the number of I/O servers, and I/O server sizes.

prep_time - Preparation time monitor element

Time in milliseconds required to prepare an SQL statement (if the activity is an SQL statement; otherwise, the value is 0).

Table 1006. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	Always collected

Table 1007. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-
Package cache	-	COLLECT BASE DATA

Usage

The prep_time monitor element indicates how much time was spent preparing the SQL statement, if this activity was an SQL statement, when the statement was first introduced to the DB2 package cache. This preparation time is not part of the activity lifetime nor does it represent time spent during a specific invocation of the statement if the statement has already been cached in the package cache prior to that invocation.

prep_time_best - Statement best preparation time monitor element

The shortest amount of time in milliseconds that was required to prepare a specific SQL statement.

Table 1008. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Dynamic SQL	dynsql	Basic

Usage

Use this value in conjunction with **prep_time_worst** to identify SQL statements that are expensive to compile.

prep_time_worst - Statement worst preparation time monitor element

The longest amount of time in milliseconds that was required to prepare a specific SQL statement.

Table 1009. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Dynamic SQL	dynsql	Basic

Usage

Use this value in conjunction with **prep_time_best** to identify SQL statements that are expensive to compile.

prev_uow_stop_time - Previous Unit of Work Completion Timestamp

This is the time the unit of work completed.

Element identifier
prev_uow_stop_time

Element type
timestamp

Table 1010. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Unit of Work, Timestamp
DCS Application	dcs_appl	Unit of Work, Timestamp

Table 1011. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Transaction	event_xact	-

Usage You may use this element with *uow_stop_time* to calculate the total elapsed time between COMMIT/ROLLBACK points, and with *uow_start_time* to calculate the time spent in the application between units of work. The time of one of the following:

- For applications currently within a unit of work, this is the time that the latest unit of work completed.
- For applications not currently within a unit of work (the application has completed a unit of work, but not yet started a new one), this is the stop time of the last unit of work that completed prior to the one that just completed. The stop time of the one just completed is indicated uow_stop_time.
- For applications within their first unit of work, this is the database connection request completion time.

priv_workspace_num_overflows - Private Workspace Overflows

The number of times that the private workspaces overflowed the bounds of its allocated memory.

Note: This monitor element has been deprecated. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

Table 1012. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 1013. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage Use this element with priv_workspace_size_top to determine whether the size of the private workspace needs to be increased to avoid overflowing. Overflows of the private workspace may cause performance degradation as well as out of memory errors from the other heaps allocated out of agent private memory.

At the database level, the element reported will be from the same private workspace as that which was reported as having the same Maximum Private Workspace size. At the application level, it is the number of overflows for the workspace of every agent that have serviced the current application.

priv_workspace_section_inserts - Private Workspace Section Inserts

Inserts of SQL sections by an application into the private workspace.

Note: This monitor element has been deprecated. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

Table 1014. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 1015. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage The working copy of executable sections are stored in the private workspace.

This counter indicates when a copy was not available and had to be inserted. At the database level, it is the cumulative total of all inserts for every application across all private workspaces in the database. At the application level, it is the cumulative total of all inserts for all sections in the private workspace for this application.

In a concentrator environment where agents are being associated with different applications, additional private workspace inserts may be required as a result of a new agent not having the required section available in its private workspace.

priv_workspace_section_lookups - Private Workspace Section Lookups

Lookups of SQL sections by an application in its agents' private workspace.

Note: This monitor element has been deprecated. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

Table 1016. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 1017. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage Each application has access to the private workspace of the agent working for it.

This counter indicates how many times the private workspace was accessed in order to locate a specific section for an application. At the database level, it is the cumulative total of all lookups for every application across all private workspaces in the database. At the application level, it is the cumulative total of all lookups for all sections in the private workspace for this application.

You can use this element in conjunction with Private Workspace Section Inserts to tune the size of the private workspace. The size of the private workspace is controlled by the applheapsz configuration parameter.

priv_workspace_size_top - Maximum Private Workspace Size

The largest size reached by the Private Workspace.

Note: This monitor element has been deprecated. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

Table 1018. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

Table 1019. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage Each agent has a private workspace that the application it is servicing has access to. This element indicates the maximum number of bytes required from a private workspace by any agent servicing it. At the database level, it is the maximum number of bytes required of all the private workspaces for all agents attached to the current database. At the application level, it is the maximum size from among all of the agents' private workspaces that have serviced the current application.

When the private workspace overflows, memory is temporarily borrowed from other entities in agent private memory. This can result in memory shortage errors from these entities or possibly performance degradation. You can reduce the chance of overflow by increasing APPLHEAPSZ.

product_name - Product Name

Details of the version of the DB2 instance that is running.

Table 1020. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

progress_completed_units - Completed Progress Work Units

The number of work units for the current phase which have been completed.

Table 1021. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	progress	Basic

The value of this element will typically increase as the utility operates. This element will always be less than or equal to *progress_total_units* (if both elements are defined).

Note:

1. This element might not be included for all utilities.
2. This element is expressed in units displayed by the *progress_work_metric* monitor element.

Usage Use this element to determine the amount of completed work within a phase. By itself, this element can be used to monitor the activity of a running utility. This element should constantly increase as the utility executes. If the *progress_completed_units* fails to increase over a long period of time then the utility might be stalled.

If *progress_total_units* is defined, then this element can be used to calculate the percentage of completed work:

```
percentage complete = progress_completed_units / progress_total_units* 100
```

progress_description - Progress Description

Describes the phase of work.

Table 1022. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	progress	Basic

Example values for the load utility include:

- DELETE
- LOAD
- REDO

Usage Use this element to obtain a general description of a phase.

progress_list_attr - Current Progress List Attributes

This element describes how to interpret a list of progress elements.

Table 1023. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	progress list	Basic

Usage

The value for this element is one of the following constants:

- SQLM_ELM_PROGRESS_LIST_ATTR_SERIAL - The elements in the list are to be interpreted as a set of serial phases meaning that completed work must equal the total work for element *n* before the completed work of element *n*+1 is first updated. This attribute is used to describe progress of a task which consists of a set of serial phases where a phase must fully complete before the next phase begins.
- SQLM_ELM_PROGRESS_LIST_ATTR_CONCURRENT - Any element in the progress list can be updated at any time.

Use this element to determine how the elements of a progress list will be updated.

progress_list_cur_seq_num - Current Progress List Sequence Number

If the utility contains multiple sequential phases, then this element displays the number of the current phase.

Table 1024. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	progress_list	Basic

Usage Use this element to determine the current phase of a multiphase utility. See "progress_seq_num - Progress Sequence Number."

progress_seq_num - Progress Sequence Number

Phase number.

Note: The phase number displays only for utilities that consist of multiple phases of execution.

Table 1025. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	progress	Basic

Usage Use this element to determine the order of phases within a multiphase utility. The utility will execute phases serially in order of increasing progress sequence numbers. The current phase of a multiphase utility can be found by matching the *progress_seq_num* with the value of *progress_list_current_seq_num*.

progress_start_time - Progress Start Time

A timestamp representing the start of the phase.

Table 1026. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	progress	Basic

Usage Use this element to determine when a phase started. This element is omitted if the phase has not yet begun.

progress_total_units - Total Progress Work Units

Total amount of work to perform in order for the phase to be complete.

Table 1027. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	progress	Basic

Some utilities might not be able to quantify the total work so they will continuously update this element. Other utilities might not be able to provide an estimate for the total work so this element might be omitted entirely.

This element is expressed in units displayed by the *progress_work_metric* monitor element.

Usage Use this element to the determine the total amount of work in the phase. Use this element with *progress_completed_units* to calculate the percentage of work completed within a phase:

```
percentage complete = progress_completed_units / progress_total_units* 100
```

progress_work_metric - Progress Work Metric

The metric for interpreting the *progress_total_units* and *progress_completed_units* elements.

Table 1028. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	progress	Basic

Example values include:

- SQLM_WORK_METRIC_BYTES
- SQLM_WORK_METRIC_EXTENTS

Note:

1. This element might not be included for all utilities.
2. Values for this element can be found in sqlmon.h

Usage Use this element to determine what *progress_total_units* and *progress_completed_units* use as their reporting metric.

pseudo_deletes - Pseudo deletes monitor element

The number of keys that have been marked pseudo deleted.

Table 1029. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

pseudo_empty_pages - Pseudo empty pages monitor element

The number of pages that have been identified as pseudo empty. Pseudo empty pages are pages where all the keys have been pseudo deleted.

Table 1030. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

Usage

Note: This monitor element does not report the current number of pseudo empty pages.

qp_query_id - Query patroller query ID monitor element

The query ID assigned to this activity by Query Patroller if the activity is a query. A query ID of 0 indicates that Query Patroller did not assign a query ID to this activity.

Important: The qp_query_id monitor element is deprecated because it is associated with Query Patroller functionality. With the new workload management features introduced in DB2 Version 9.5, Query Patroller and its related components have been deprecated in Version 9.7 and might be removed in a future release.

Table 1031. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

query_card_estimate - Query Number of Rows Estimate

An estimate of the number of rows that will be returned by a query.

Table 1032. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement
DCS Statement	dcs_stmt	Statement

Table 1032. Snapshot Monitoring Information (continued)

Snapshot Level	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

Usage This estimate by the SQL compiler can be compared with the run time actuals.

This element also returns information for the following SQL statements when you are monitoring DB2 Connect.

- INSERT, UPDATE, and DELETE

  Indicates the number of rows affected.

- PREPARE

  Estimate of the number of rows that will be returned. Only collected if the DRDA server is DB2 Database for Linux, UNIX, and Windows, DB2 for VM and VSE, or DB2 for OS/400®.

- FETCH

  Set to the number of rows fetched. Only collected if the DRDA server is DB2 for OS/400.

If information is not collected for a DRDA server, then the element is set to zero.

query_cost_estimate - Query cost estimate monitor element

Estimated cost for a query, as determined by the SQL compiler. This value is reported in timerons.

Table 1033. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	Always collected

Table 1034. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement
DCS Statement	dcs_stmt	Statement
Activities	event_activity	-
Package cache	-	COLLECT BASE DATA

Usage

This monitor element allows correlation of actual run-time with the compile-time estimates.

This element also returns information for the following SQL statements when you are monitoring DB2 Connect.

- PREPARE

  Represents the relative cost of the prepared SQL statement.
- FETCH

  Contains the length of the row retrieved. Only collected if the DRDA server is DB2 for OS/400.

If information is not collected for a DRDA server, then the element is set to zero.

Note: If the DRDA server is DB2 for OS/390® and z/OS, this estimate could be higher than 2**32 - 1 (the maximum integer number that can be expressed through an unsigned long variable). In that case, the value returned by the monitor for this element will be 2**32 - 1.

queue_assignments_total - Queue assignments total monitor element

The number of times any connection or activity was assigned to this threshold queue since the last reset.

Table 1035. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_qstats	-

Usage

This element can be used to determine the number of times any connection or activity was queued in this particular queue in a given period of time determined by the statistics collection interval. This can help to determine the effectiveness of queuing thresholds.

queue_size_top - Queue size top monitor element

Highest queue size that has been reached since the last reset.

Table 1036. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_qstats	-

Usage

Use this element to gauge the effectiveness of queuing thresholds and to detect when queuing is excessive.

queue_time_total - Queue time total monitor element

Sum of the times spent in the queue for all connections or activities placed in this queue since the last reset. Units are milliseconds.

Table 1037. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_qstats	-

This element is used to gauge the effectiveness of queuing thresholds and to detect when queuing is excessive.

Usage notes

queue_time_total is not reset at the end of a statistic collection interval. If **queue_time_total** is used over multiple intervals, it can be greater than the product of **wlm_collect_int** and **queue_size_top**.

quiescer_agent_id - Quiescer Agent Identification

Agent ID of the agent holding a quiesce state.

Element identifier
quiescer_agent_id

Element type
information

Table 1038. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_quiescer	Basic

Usage Use this element in conjunction with quiescer_auth_id to determine who is responsible for quiescing a table space.

quiescer_auth_id - Quiescer User Authorization Identification

Authorization ID of the user holding a quiesce state.

Element identifier
quiescer_auth_id

Element type
information

Table 1039. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_quiescer	Basic

Usage Use this element to determine who is responsible for quiescing a table space.

quiescer_obj_id - Quiescer Object Identification

The object ID of the object that causes a table space to be quiesced.

Element identifier
quiescer_obj_id

Element type
information

Table 1040. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_quiescer	Basic

Usage Use this element in conjunction with quiescer_ts_id and quiescer_auth_id to determine who is responsible for quiescing a table space. The value of this element matches a value from column TABLEID of view SYSCAT.TABLES.

quiescer_state - Quiescer State

The type of quiesce being done (for example, "SHARE", "INTENT TO UPDATE", or "EXCLUSIVE").

Element identifier
quiescer_state

Element type
information

Table 1041. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_quiescer	Basic

Usage The value of this element matches the value of constants SQLB_QUIESCED_SHARE, SQLB_QUIESCED_UPDATE, or SQLB_QUIESCED_EXCLUSIVE from sqlutil.h.

quiescer_ts_id - Quiescer Table Space Identification

The table space ID of the object that causes a table space to be quiesced.

Element identifier
quiescer_ts_id

Element type
information

Table 1042. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_quiescer	Basic

Usage Use this element in conjunction with quiescer_obj_id and quiescer_auth_id to determine who is responsible for quiescing a table space. The value of this element matches a value from column TBSPACEID of view SYSCAT.TABLES.

range_adjustment - Range Adjustment

This value represents the offset into the container array in which a range actually starts.

Table 1043. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_range	Basic

Usage This element is applicable only to a DMS table space.

range_container_id - Range Container

An integer that uniquely defines a container within a range.

Table 1044. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_range	Basic

Usage This element is applicable only to a DMS table space.

range_end_stripe - End Stripe

This value represents the number of the last stripe in a range.

Table 1045. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_range	Basic

Usage This element is applicable only to a DMS table space.

range_max_extent - Maximum Extent in Range

This value represents the maximum extent number that is mapped by a range.

Table 1046. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_range	Basic

Usage This element is applicable only to a DMS table space.

range_max_page_number - Maximum Page in Range

This value represents the maximum page number that is mapped by a range.

Table 1047. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_range	Basic

Usage This element is applicable only to a DMS table space.

range_num_containers - Number of Containers in Range

This value represents the number of containers in the current range.

Table 1048. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_range	Basic

Usage This element is applicable only to a DMS table space.

range_number - Range Number

This value represents the number of a range within the table space map.

Table 1049. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_range	Basic

Usage This element is applicable only to a DMS table space.

range_offset - Range Offset

The offset from stripe 0 of the beginning of the stripe set to which a range belongs.

Table 1050. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_range	Basic

Usage This element is applicable only to a DMS table space.

range_start_stripe - Start Stripe

This value represents the number of the first stripe in a range.

Table 1051. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_range	Basic

Usage This element is applicable only to a DMS table space.

range_stripe_set_number - Stripe Set Number

This value represents the stripe set in which a range resides.

Table 1052. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_range	Basic

Usage This element is applicable only to a DMS table space.

reclaimable_space_enabled - Reclaimable space enabled indicator monitor element

If the table space is enabled for reclaimable storage, then this monitor element returns a value of 1. Otherwise, it returns a value of 0.

Table 1053. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

rej_curs_blk - Rejected Block Cursor Requests

The number of times that a request for an I/O block at server was rejected and the request was converted to non-blocked I/O.

Element identifier
rej_curs_blk

Element type
counter

Table 1054. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

Table 1055. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

Usage If there are many cursors blocking data, the communication heap may become full. When this heap is full, an error is not returned. Instead, no more I/O blocks are allocated for blocking cursors. If cursors are unable to block data, performance can be affected.

If a large number of cursors were unable to perform data blocking, you may be able to improve performance by:

- Increasing the size of the *query_heap* database manager configuration parameter.

rem_cons_in - Remote Connections To Database Manager

The current number of connections initiated from remote clients to the instance of the database manager that is being monitored.

Table 1056. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

Usage

Shows the number of connections from remote clients to databases in this instance. This value will change frequently, so you may need to sample it at specific intervals over an extended period of time to get a realistic view of system usage. This number does not include applications that were initiated from the same instance as the database manager.

When used in conjunction with the local_cons monitor element, these elements can help you adjust the setting of the **max_coordagents** and **max_connections** configuration parameters.

rem_cons_in_exec - Remote Connections Executing in the Database Manager

The number of remote applications that are currently connected to a database and are currently processing a unit of work within the database manager instance being monitored.

Table 1057. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

Usage

This number can help you determine the level of concurrent processing occurring on the database manager. This value will change frequently, so you may need to sample it at specific intervals over an extended period of time to get a realistic view of system usage. This number does not include applications that were initiated from the same instance as the database manager.

When used in conjunction with the local_cons_in_exec monitor element, this element can help you adjust the setting of the **max_coordagents** configuration parameter.

If **max_coordagents** is set to AUTOMATIC, then you do not need to make any adjustments. If it is not set to AUTOMATIC and if the sum of rem_cons_in_exec and local_cons_in_exec is close to **max_coordagents**, you should increase the value of **max_coordagents**.

remote_lock_time - Remote Lock Time

This element contains the aggregate amount of time, in milliseconds, that this data source spends in a remote lock from all applications or a single application running on this federated server instance since the start of the federated server instance or the last reset of the database monitor counters, whichever is latest. The response time is measured as the difference between the time the federated server submits a remote lock to the data source, and the time the federated server releases a remote lock at the data source.

Table 1058. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Timestamp
Application	appl_remote	Timestamp

For snapshot monitoring, this counter can be reset.

Usage

Use this element to determine how much actual time is spent at this data source in a remote lock.

remote_locks - Remote Locks

This element contains a count of the total number of remote locks that the federated server has called at this data source on behalf of any application since the start of the federated server instance or the last reset of the database monitor counters, whichever is latest.

Table 1059. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

Usage Use this element to determine how many remote locks were made remotely at the data source.

remote_member - Remote member monitor element

The numeric identifier for the database member to which data was sent or from which data was received, using the fast communication manager (FCM).

Table 1060. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_FCM_CONNECTION_LIST - Get details for all FCM connections	Always collected

Usage

All the metrics returned by the MON_GET_FCM_CONNECTION_LIST table function apply to the FCM connection between the members described in the **member** and **remote_member** monitor elements.

reorg_completion - Reorganization Completion Flag

Table reorganization success indicator, which includes the reclamation of extents from a multidimensional clustering (MDC) table. For partitioned tables, this also indicates the completion status for the data partition.

Table 1061. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

Usage This element will have a value of 0 if a table or data partition reorganize operation is successful. If a table or data partition reorganize operation is unsuccessful, this element will have a value of -1. Success and failure values are defined in sqlmon.h as follows:

- Success: SQLM_REORG_SUCCESS
- Failure: SQLM_REORG_FAIL

In the case of an unsuccessful table reorganization, see the history file for any diagnostic information, including warnings and errors. This data can be accessed by using the LIST HISTORY command. For partitioned tables,

the completion status is indicated per data partition. If index recreate fails on a partitioned table, the failed status is updated on all data partitions. See the administration notification log for further diagnostic information.

reorg_current_counter - Reorganize Progress

A unit of progress that indicates the amount of reorganization that has been completed. The amount of progress this value represents is relative to the value of reorg_max_counter, which represents the total amount of table reorganization that is to be done.

Table 1062. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

Usage

You can determine the percentage of table reorganization that has been completed using the following formula:

```
table reorg progress = reorg_current_counter / reorg_max_counter * 100
```

reorg_end - Table Reorganize End Time

The end time of a table reorganization including a reorganization to reclaim extents from a multidimensional clustering (MDC) table. For partitioned tables, this will also indicate the end time for each data partition reorganization.

Table 1063. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

reorg_index_id - Index Used to Reorganize the Table

The index being used to reorganize the table.

Table 1064. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

reorg_long_tbspc_id - Table Space Where Long Objects are Reorganized monitor element

The table space in which any long objects (LONG VARCHAR or LOB data) will be reorganized. For partitioned tables, this is the table space in which each partition's LONG VARCHAR and LOB will be reorganized.

Table 1065. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

reorg_max_counter - Total Amount of Reorganization

A value that indicates the total amount of work to be done in a reorganization including a reorganization to reclaim extents from multidimensional clustering (MDC) tables.. This value can be used with reorg_current_counter, which represents the amount of work completed, to determine the progress of a reorganization.

Table 1066. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

reorg_max_phase - Maximum Reorganize Phase

The maximum number of reorganization phases that will occur during reorganization processing. This applies to classic (offline) reorganizations only. The range of values is 2 to 4 ([SORT], BUILD, REPLACE,[INDEX_RECREATE]). The value could also indicate the total amount of work to be done in a reorganization to reclaim extents from a multidimensional clustering (MDC) table. When such a reorganization occurs, this value is 3 (SCAN, DRAIN, RELEASE).

Table 1067. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

reorg_phase - Table reorganization phase monitor element

Indicates the reorganization phase of the table. For partitioned tables, this will also indicate the reorganization phase for each data partition. This applies to offline table reorganizations only.

Table 1068. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

Usage

For partitioned tables, the reorganization occurs on a data partition by data partition basis. For classic table reorganization, the following phases are possible (phases are listed with their corresponding defines from the`sqlmon.h` file):

- Sort: SQLM_REORG_SORT
- Build: SQLM_REORG_BUILD
- Replace: SQLM_REORG_REPLACE
- Index Recreate: SQLM_REORG_INDEX_RECREATE
- Dictionary Build: SQLM_REORG_DICT_SAMPLE

For partitioned tables, the Index Recreate phase for partitioned indexes (if any) might be directly entered after the replace phase for that data partition. The **`reorg_phase`** element will indicate the Index Recreate phase only after the successful completion of all prior phases on every data partition.

During XDA object compression, the XML data reorganization phase involves reorganizing the XML storage object of the table. The XML dictionary build phase involves attempting to create a compression dictionary for the XML storage object. For XDA object compression, the following two phases are possible:

- XML Reorg: SQLM_REORG_XML_DATA
- XML Dictionary Build: SQLM_REORG_XML_DICT_SAMPLE

For partitioned tables, where reclamation of extents is being performed, the following phases are possible:

- Scan: SQLM_REORG_SCAN
- Drain: SQLM_REORG_DRAIN
- Release: SQLM_REORG_RELEASE

reorg_phase_start - Reorganize Phase Start Time

The start time of a phase of table reorganization or reclaim reorganization. For partitioned tables, this will also indicate the start time of a reorganization phase for each data partition. During the index recreate phase, data groups for all data partitions are updated at the same time for nonpartitioned indexes.

Table 1069. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

reorg_rows_compressed - Rows Compressed

Number of rows compressed in the table during reorganization.

Table 1070. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

Usage A running count of the number of rows compressed in the table during reorganization. Some records may never be compressed (if the record size is less than the minimum record length).

It is important to note that this row count does not measure the effectiveness of data compression. It only displays the number of records meeting compression criteria.

reorg_rows_rejected_for_compression - Rows Rejected for Compression

Number of rows that were not compressed during reorganization due to the record length being less than or equal to the minimum record length.

Table 1071. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

Usage A record will not be compressed if it is less than or equal to the minimum

record length. The number of rows rejected reflects a running count for these records that fail to meet this compression requirement.

reorg_start - Table Reorganize Start Time

The start time of a table reorganization including a reorganization to reclaim extents from a multidimensional clustering (MDC) table. For partitioned tables, this will also indicate the start time for each data partition reorganization.

Table 1072. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

reorg_status - Table Reorganize Status

The status of an in-place (online) table or a data partition level reorganization. This is not applicable to classic (offline) table reorganizations.

Table 1073. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

Usage An in-place table or data partition reorganization can be in one of the following states (states are listed with their corresponding defines from sqlmon.h):

- Started/Resumed: SQLM_REORG_STARTED
- Paused: SQLM_REORG_PAUSED
- Stopped: SQLM_REORG_STOPPED
- Completed: SQLM_REORG_COMPLETED
- Truncate: SQLM_REORG_TRUNCATE

An inplace table or data partition reorganization to reclaim extents can be in one of the following states:

- Started: SQLM_REORG_STARTED
- Stopped: SQLM_REORG_STOPPED
- Completed: SQLM_REORG_COMPLETED

reorg_tbspc_id - Table Space Where Table or Data partition is Reorganized

The table space in which the table will be reorganized. For partitioned tables, this indicates the table space where each data partition is reorganized.

Table 1074. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

reorg_type - Table Reorganize Attributes

Table reorganize attribute settings.

Table 1075. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

Usage The following are possible attribute settings. Each attribute setting is based upon a bit flag value defined in `db2ApiDf.h`.

- Allow Write Access: DB2REORG_ALLOW_WRITE
- Allow Read Access: DB2REORG_ALLOW_READ
- Allow No Access: DB2REORG_ALLOW_NONE
- Recluster Via Index Scan: DB2REORG_INDEXSCAN
- Reorg Long Field LOB Data: DB2REORG_LONGLOB
- No Table Truncation: DB2REORG_NOTRUNCATE_ONLINE
- Replace Compression Dictionary: DB2REORG_RESET_DICTIONARY
- Keep Compression Dictionary: DB2REORG_KEEP_DICTIONARY
- Reclaim Extents: DB2REORG_RECLAIM_EXTS

In addition to the preceding attribute settings, the following attributes are listed in the CLP output of the GET SNAPSHOT FOR TABLES command. These attribute settings are based on the values of other attribute settings or table reorganize monitor elements.

- Reclustering: If the value of the reorg_index_id monitor element is non-zero, then the table reorganize operation has this attribute.
- Reclaiming: If the value of the reorg_index_id monitor element is zero, then the table reorganize operation has this attribute.
- Inplace Table Reorg: If the reorg_status monitor element has a value that is not null, then the in-place (online) reorganization method is in use.
- Table Reorg: If the reorg_phase monitor element has a value that is not null, then the classic (offline) reorganization method is in use.
- Recluster Via Table Scan: If the DB2REORG_INDEXSCAN flag is not set, then the table reorganize operation has this attribute.
- Reorg Data Only: If the DB2REORG_LONGLOB flag is not set, then the table reorganize operation has this attribute.

reorg_xml_regions_compressed – XML regions compressed monitor element

Number of XML regions that were compressed during the table reorganization process.

Table 1076. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

reorg_xml_regions_rejected_for_compression – XML regions rejected for compression monitor element

Number of XML regions that were not compressed during the table reorganization process.

Table 1077. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

req_agent_tid - Thread identifier for agent waiting to acquire lock monitor element

Thread identifier of the agent or system entity that is waiting to acquire the lock.

Table 1078. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_APPL_LOCKWAIT table function - get information about locks for which an application is waiting	

req_application_handle - Identifier for application waiting to acquire lock monitor element

System-wide unique ID for the application that is waiting to acquire the lock.

Table 1079. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_APPL_LOCKWAIT table function - get information about locks for which an application is waiting	

req_executable_id - Identifier for statement section waiting to acquire lock monitor element

The binary token generated on the data server that uniquely identifies the SQL statement section which is waiting to acquire a lock. For non-SQL activities, a 0-length string value is returned.

Table 1080. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_APPL_LOCKWAIT table function - get information about locks for which an application is waiting	

req_member - Member of application waiting to acquire lock monitor element

Database member where the application waiting to acquire this lock is located.

Table 1081. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_APPL_LOCKWAIT table function - get information about locks for which an application is waiting	

request_exec_time_avg - Request execution time average monitor element

Arithmetic mean of the execution times for requests associated with this service subclass since the last reset. If the internally tracked average has overflowed, the value -2 is returned. This monitor element returns -1 when COLLECT AGGREGATE REQUEST DATA for the service subclass is set to NONE. Units are milliseconds.

When you remap activities between service subclasses with a REMAP ACTIVITY action, the request_exec_time_avg mean counts the partial request in each subclass involved in remapping.

Table 1082. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-

Usage

Use this statistic to quickly understand the average amount of time that is spent processing each request on a database partition in this service subclass.

This average can also be used to determine whether or not the histogram template used for the request execution time histogram is appropriate. Compute the average request execution time from the request execution time histogram. Compare the computed average with this monitor element. If the computed average deviates from the true average reported by this monitor element, consider altering the histogram template for the request execution time histogram, using a set of bin values that are more appropriate for your data.

rf_log_num - Log Being Rolled Forward

The log being processed.

Element identifier
rf_log_num

Element type
information

Table 1083. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	rollforward	Basic

Usage If a rollforward is in progress, this element identifies the log involved.

rf_status - Log Phase

The status of the recovery.

Element identifier
rf_status

Element type
information

Table 1084. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	rollforward	Basic

Usage This element indicates the progression of a recovery. It indicates if the recovery is in an undo (rollback) or redo (rollforward) phase.

rf_timestamp - Rollforward Timestamp

The timestamp of the last committed transaction..

Element identifier
rf_timestamp

Element type
timestamp

Table 1085. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	rollforward	Timestamp

Usage If a rollforward is in progress, this is the timestamp of the last committed transaction processed by rollforward recovery. This is an indicator of how far the rollforward operation has progressed.

rf_type - Rollforward Type

The type of rollforward in progress.

Element identifier
rf_type

Element type
information

Table 1086. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	rollforward	Basic

Usage An indicator of whether recovery is happening at a database or table space level.

rollback_sql_stmts - Rollback Statements Attempted

The total number of SQL ROLLBACK statements that have been attempted.

Element identifier
rollback_sql_stmts

Element type
counter

Table 1087. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Database	dbase_remote	Basic
Application	appl	Basic
Application	appl_remote	Basic
DCS Database	dcs_dbase	Basic
DCS Application	dcs_appl	Basic

For snapshot monitoring, this counter can be reset.

Table 1088. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage A rollback can result from an application request, a deadlock, or an error situation. This element **only** counts the number of rollback statements issued from applications.

At the application level, this element can help you determine the level of database activity for the application and the amount of conflict with other applications. At the database level, it can help you determine the amount of activity in the database and the amount of conflict between applications on the database.

Note: You should try to minimize the number of rollbacks, since higher rollback activity results in lower throughput for the database.

It may also be used to calculate the total number of units of work, by calculating the sum of the following:

```
  commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks
```

rolled_back_agent_id - Rolled Back Agent

Agent that was rolled back when a deadlock occurred.

Element identifier
rolled_back_agent_id

Element type
information

Table 1089. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	event_deadlock	-

Usage A system administrator can use this information to determine which application did not complete its updates, and determine which applications should be restarted.

rolled_back_appl_id - Rolled Back Application

Application id that was rolled back when a deadlock occurred.

Element identifier
rolled_back_appl_id

Element type
information

Table 1090. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	event_deadlock	-

Usage A system administrator can use this information to determine which application did not complete its updates, and determine which applications should be restarted.

rolled_back_participant_no - Rolled back application participant monitor element

The participant number identifying the rolled back application.

Table 1091. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks[1]	event_deadlock	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

A system administrator can use this information to determine which application did not complete its updates, and determine which application should be started.

rolled_back_sequence_no - Rolled Back Sequence Number

The sequence number of the application that was rolled back when a deadlock occurred.

Element identifier
rolled_back_sequence_no

Element type
information

Table 1092. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	event_deadlock	-

Usage A system administrator can use this information to determine which application did not complete its updates, and determine which applications should be restarted.

root_node_splits - Root node splits monitor element

Number of times the root node of the index was split during an insert operation.

Table 1093. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

routine_id - Routine ID monitor element

This monitor element is a unique routine identifier. It returns zero if the activity is not part of a routine.

Table 1094. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 1095. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activitystmt	-
Unit of work	Reported in the package list.	-

Usage

The value of this element matches a value from column ROUTINEID of view SYSCAT.ROUTINES. When the activity is part of a routine declared within another SQL PL routine, the value of this element is the ROUTINEID of the outer routine.

rows_deleted - Rows deleted monitor element

This is the number of row deletions attempted.

Table 1096. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLE table function - Get table metrics	Always collected

Table 1097. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Database	dbase_remote	Basic
Application	appl	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

Table 1098. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage You can use this element to gain insight into the current level of activity within the database.

This count does not include the attempts counted in the **int_rows_deleted** monitor element.

rows_fetched - Rows fetched monitor element

The number of rows read from the table.

This monitor element is an alias of the **rows_read** monitor element.

Note: This monitor element reports only the values for the database partition for which this information is recorded. In partitioned database environments, these values may not reflect the correct totals for the whole activity.

Table 1099. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	Statement

Usage

See the **rows_read** monitor element for details.

rows_inserted - Rows inserted monitor element

The number of row insertions attempted.

Table 1100. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLE table function - Get table metrics	Always collected

Table 1101. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Database	dbase_remote	Basic
Application	appl	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

Table 1102. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage You can use this element to gain insight into the current level of activity within the database.

In a federated system, multiple rows can be inserted, per INSERT statement, because the federated server can push INSERT FROM SUBSELECT to the data source, when appropriate.

This count does not include the attempts counted in the **int_rows_inserted** monitor element.

rows_modified - Rows modified monitor element

The number of rows inserted, updated, or deleted.

This monitor element is an alias of the **rows_written** monitor element.

Table 1103. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 1103. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 1104. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Activities	event_activity	Statement
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

See the **rows_written** monitor element for details.

rows_read - Rows read monitor element

The number of rows read from the table.

Table 1105. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLE table function - Get table metrics	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 1106. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Table	table	Table
Application	appl	Basic
Application	stmt	Basic
Application	subsection	Statement
Dynamic SQL	dynsql	Statement

For snapshot monitoring, this counter can be reset.

Table 1107. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Connection	event_conn	-
Tables	event_table	-
Statements	event_stmt	-
Transactions	event_xact	-
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

This element helps you identify tables with heavy usage for which you may want to create additional indexes. To avoid the maintenance of unnecessary indexes, use the SQL EXPLAIN statement to determine if the package uses an index.

This count is *not* the number of rows that were returned to the calling application. Rather, it is the number of rows that had to be read in order to return the result set. For example, the following statement returns one row to the application, but many rows are read to determine the average salary:

```
SELECT AVG(SALARY) FROM USERID.EMPLOYEE
```

This count includes the value in the **overflow_accesses** monitor element. Additionally, this count does not include any index accesses. That is, if an access plan uses index access only and the table is not touched to look at the actual row, then the value of the **rows_read** monitor element is not incremented.

rows_returned - Rows returned monitor element

The number of rows that have been selected and returned to the application. This element has a value of 0 for partial activity records (for example, if an activity is collected while it is still executing or when a full activity record could not be written to the event monitor due to memory limitations).

This monitor element is an alias of the **fetch_count** monitor element.

Table 1108. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 1109. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Activities	event_activity	-
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

Table 1109. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

This element can be used to help determine thresholds for rows returned to the application or can be used to verify that such a threshold is configured correctly and doing its job.

rows_returned_top - Actual rows returned top monitor element

The high watermark for the actual rows returned of DML activities at all nesting levels in a service class or work class. For service classes, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service class is set to NONE. For work classes, this monitor element returns -1 if no COLLECT AGGREGATE ACTIVITY DATA work action is specified for the work class. For workloads, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the workload is set to NONE.

For service classes, when you remap activities between service subclasses with a REMAP ACTIVITY action, only the rows_returned_top high watermark of the service subclass where an activity completes is updated. High watermarks of service subclasses an activity is mapped to but does not complete in are unaffected.

Table 1110. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wcstats	-
Statistics	event_wlstats	-

Usage

Use this element to know the highest DML activity actual rows returned reached on a partition for a service class, workload, or work class in the time interval collected.

rows_selected - Rows Selected

This is the number of rows that have been selected and returned to the application.

Table 1111. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Database	dbase_remote	Basic
Application	appl	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

Table 1112. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage You can use this element to gain insight into the current level of activity within the database.

This element does not include a count of rows read for actions such as COUNT(*) or joins.

For a federated system;, you can compute the average time to return a row to the federated server from the data source:

```
average time = rows returned / aggregate query response time
```

You can use these results to modify CPU speed or communication speed parameters in SYSCAT.SERVERS. Modifying these parameters can impact whether the optimizer does or does not send requests to the data source.

Note: This element is collected at the dcs_dbase and dcs_appl snapshot monitor logical data groups if the gateway being monitored is at DB2 database version 7.2 or lower.

rows_updated - Rows updated monitor element

This is the number of row updates attempted.

Table 1113. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLE table function - Get table metrics	Always collected

Table 1114. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Database	dbase_remote	Basic
Application	appl	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

Table 1115. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage You can use this element to gain insight into the current level of activity within the database.

This value does not include updates counted in the **int_rows_updated** monitor element. However, rows that are updated by more than one update statement are counted for each update.

rows_written - Rows Written

This is the number of rows changed (inserted, deleted or updated) in the table.

Table 1116. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table	Basic
Application	appl	Basic
Application	stmt	Basic
Application	subsection	Statement
Dynamic SQL	dynsql	Statement

For snapshot monitoring, this counter can be reset.

Table 1117. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-
Tables	event_table	-
Statements	event_stmt	-
Transactions	event_xact	-

Usage A high value for table-level information indicates there is heavy usage of the table and you may want to use the Run Statistics (RUNSTATS) utility to maintain efficiency of the packages used for this table.

For application-connections and statements, this element includes the number of rows inserted, updated, and deleted in temporary tables.

At the application, transaction, and statement levels, this element can be useful for analyzing the relative activity levels, and for identifying candidates for tuning.

rqsts_completed_total - Total requests completed monitor element

The total number of requests executed, including both application and internal requests. For service subclasses, this monitor element is only updated where the request completes. If the request moved between different service subclasses, it is not counted twice.

Table 1118. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 1118. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1119. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

sc_work_action_set_id - Service class work action set ID monitor element

If this activity has been categorized into a work class of service class scope, this monitor element displays the ID of the work action set associated with the work class set to which the work class belongs. Otherwise, this monitor element displays the value of 0.

Table 1120. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
WLM_GET_ACTIVITY_DETAILS_COMPLETE table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 1121. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

Usage

This element can be used with the **sc_work_class_id** element to uniquely identify the service class work class of the activity, if one exists.

sc_work_class_id - Service class work class ID monitor element

If this activity has been categorized into a work class of service class scope, this monitor element displays the ID of the work class assigned to this activity. Otherwise, this monitor element displays the value of 0.

Table 1122. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
WLM_GET_ACTIVITY_DETAILS_COMPLETE table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 1123. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

Usage

This element can be used with the **sc_work_action_set_id** element to uniquely identify the service class work class of the activity, if one exists.

sec_log_used_top - Maximum Secondary Log Space Used

The maximum amount of secondary log space used (in bytes).

Table 1124. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Table 1125. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage You may use this element in conjunction with *sec_logs_allocated* and *tot_log_used_top* to show your current dependency on secondary logs. If this value is high, you may need larger log files, or more primary log files, or more frequent COMMIT statements within your application.

As a result, you may need to adjust the following configuration parameters:

- logfilsiz
- logprimary
- logsecond
- logretain

The value will be zero if the database does not have any secondary log files. This would be the case if there were none defined.

Note: While the database system monitor information is given in bytes, the configuration parameters are set in pages, which are each 4K bytes.

sec_logs_allocated - Secondary Logs Allocated Currently

The total number of secondary log files that are currently being used for the database.

Table 1126. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Usage You may use this element in conjunction with *sec_log_used_top* and *tot_log_used_top* to show your current dependency on secondary logs. If this value is consistently high, you may need larger log files, or more primary log files, or more frequent COMMIT statements within your application.

As a result, you may need to adjust the following configuration parameters:

- logfilsiz
- logprimary
- logsecond
- logretain

section_actuals - Section actuals monitor element

A binary string generated at the data server containing runtime statistics for a section that was executed. If section capture or actuals collection are not enabled, the value is a 0 length string. For non-SQL activities (for example, LOAD) the value is a 0 length string.

Table 1127. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

Usage

The data collected in the **section_actuals** monitor element or per connection using **WLM_SET_CONN_ENV** is used when a section explain is performed using the EXPLAIN_FROM_ACTIVITY stored procedure. This data is used during EXPLAIN processing to populate the EXPLAIN_ACTUALS explain table and represents the runtime statistics for the operators in the access plan.

Note:

- Section actuals are only available if they have been enabled (set to BASE) using the **section_actuals** database configuration parameter or if they have been enabled for a particular application using the WLM_SET_CONN_ENV stored procedure. For more information describing the stored procedure, see WLM_SET_CONN_ENV
- The **section_actuals** setting specified by the WLM_SET_CONN_ENV procedure for an application takes effect immediately.

section_env - Section environment monitor element

A blob that contains the section for an SQL statement. It is the actual section contents, that is the executable form of the query plan.

Table 1128. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activitystmt	-
Package cache	-	COLLECT DETAILED DATA

Usage

Use this element with the section explain procedures to explain the statement and view the access plan for the statement.

section_number - Section number monitor element

The internal section number in the package for a static SQL statement.

Table 1129. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	Always collected

Table 1130. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement
DCS Statement	dcs_stmt	Statement

Table 1131. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details[1]	event_detailed_dlconn	-
Statements	event_stmt	-
Activities	event_activitystmt	-
Package cache	-	COLLECT BASE DATA

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

For static SQL statements, you can use this element along with **creator**, **package_version_id**, and **package_name** monitor elements to query the SYSCAT.STATEMENTS system catalog table and obtain the static SQL statement text, using the sample query as follows:

```
SELECT SEQNO, SUBSTR(TEXT,1,120)
       FROM SYSCAT.STATEMENTS
       WHERE PKGNAME   = 'package_name' AND
             PKGSCHEMA = 'creator'      AND
             VERSION = 'package_version_id' AND
             SECTNO    = section_number
       ORDER BY SEQNO
```

Note: Exercise caution in obtaining static statement text, because this query against the system catalog table could cause lock contention. Whenever possible, only use this query when there is little other activity against the database.

section_type - Section type indicator monitor element

Indicates whether the SQL statement section is dynamic or static.

Table 1132. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	Always collected

Table 1133. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Package cache	-	COLLECT BASE DATA

Usage

The possible values for this monitor element are the following:

- D: dynamic
- S: static

select_sql_stmts - Select SQL Statements Executed

The number of SQL SELECT statements that were executed.

Table 1134. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Database	dbase_remote	Basic
Table Space	tablespace	Basic
Application	appl	Basic

Table 1134. Snapshot Monitoring Information (continued)

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

Table 1135. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage You can use this element to determine the level of database activity at the application or database level.

You can also use the following formula to determine the ratio of SELECT statements to the total statements:

```
  select_sql_stmts
/ ( static_sql_stmts
  + dynamic_sql_stmts )
```

This information can be useful for analyzing application activity and throughput.

select_time - Query Response Time

This element contains the aggregate amount of time, in milliseconds, that it has taken this data source to respond to queries from all applications or a single application running on this federated server instance since the start of the federated server instance or the last reset of the database monitor counters, whichever is latest.

Note: Due to query blocking, not all attempts by the federated server to retrieve a row result in communication processing; the request to get the next row can potentially be satisfied from a block of returned rows. As a result, the aggregate query response time does not always indicate processing at the data source, but it usually indicates processing at either the data source or client.

Table 1136. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Timestamp
Application	appl_remote	Timestamp

For snapshot monitoring, this counter can be reset.

Usage

Use this element to determine how much actual time is spent waiting for data from this data source. This can be useful in capacity planning and tuning the CPU speed and communication rates in SYSCAT.SERVERS. Modifying these parameters can impact whether the optimizer does or does not send requests to the data source.

The response time is measured as the difference in time between the time the federated server requests a row from the data source, and the time the row is available for the federated server to use.

sequence_no - Sequence number monitor element

This identifier is incremented whenever a unit of work ends (that is, when a COMMIT or ROLLBACK terminates a unit of work). Together, the **appl_id** and **sequence_no** uniquely identify a transaction.

Table 1137. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_id_info	Basic
DCS Application	dcs_appl_info	Basic

Table 1138. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-
Connections	event_connheader	-
Statements	event_stmt	-
Transactions	event_xact	-
Deadlocks	event_dlconn	-
Deadlocks with Details	event_detailed_dlconn	-
Deadlocks with Details History	event_detailed_dlconn	-
Deadlocks with Details History	event_stmt_history	-
Deadlocks with Details History Values	event_detailed_dlconn	-
Deadlocks with Details History Values	event_stmt_history	-

sequence_no_holding_lk - Sequence Number Holding Lock

The sequence number of the application that is holding a lock on the object that this application is waiting to obtain.

Element identifier
sequence_no_holding_lk

Element type
information

Table 1139. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic
Lock	appl_lock_list	Basic

Table 1140. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	event_dlconn	-
Deadlocks with Details	event_detailed_dlconn	-

Usage This identifier is used in tandem with appl_id to uniquely identify a transaction that is holding a lock on the object that this application is waiting to obtain.

server_db2_type - Database Manager Type at Monitored (Server) Node

Identifies the type of database manager being monitored.

Table 1141. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	collected	Basic

Usage It contains one of the following types of configurations for the database manager:

API Symbolic Constant
Command Line Processor Output

sqlf_nt_server
Database server with local and remote clients

sqlf_nt_stand_req
Database server with local clients

The API symbolic constants are defined in the include file *sqlutil.h*.

server_instance_name - Server Instance Name

The name of the database manager instance for which the snapshot was taken.

Element identifier
server_instance_name

Element type
information

Table 1142. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	collected	Basic

Table 1143. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Event Log Header	event_log_header	-

Usage If more than one instance of the database manager is present on the same system, this data item is used to uniquely identify the instance for which the snapshot call was issued. This information can be useful if you are

saving your monitor output in a file or database for later analysis, and you need to differentiate the data from different instances of the database manager.

server_platform - Server Operating System

The operating system running the database server.

Element identifier
server_platform

Element type
information

Table 1144. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Table 1145. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage This element can be used for problem determination for remote applications. Values for this field can be found in the header file *sqlmon.h*.

server_prdid - Server Product/Version ID

The product and version that is running on the server.

Table 1146. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	collected	Basic

Table 1147. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Event Log Header	event_log_header	-

Usage It is in the form PPPVVRRM, where:

PPP is SQL

VV identifies a 2-digit version number (with high-order 0 in the case of a 1-digit version)

RR identifies a 2-digit release number (with high-order 0 in the case of a 1-digit release)

M identifies a 1-character modification level (0-9 or A-Z)

server_version - Server Version

The version of the server returning the information.

Table 1148. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	collected	Basic

Usage

This field identifies the level of the database server collecting database system monitor information. This allows applications to interpret the data based on the level of the server returning the data. Valid values are:

SQLM_DBMON_VERSION1
Data was returned by DB2 Version 1

SQLM_DBMON_VERSION2
Data was returned by DB2 Version 2

SQLM_DBMON_VERSION5
Data was returned by DB2 Universal Database™ Version 5

SQLM_DBMON_VERSION5_2
Data was returned by DB2 Universal Database Version 5.2

SQLM_DBMON_VERSION6
Data was returned by DB2 Universal Database Version 6

SQLM_DBMON_VERSION7
Data was returned by DB2 Universal Database Version 7

SQLM_DBMON_VERSION8
Data was returned by DB2 Universal Database Version 8

SQLM_DBMON_VERSION9
Data was returned by DB2 Database for Linux, UNIX, and Windows Version 9

SQLM_DBMON_VERSION9_5
Data was returned by DB2 Database for Linux, UNIX, and Windows Version 9.5

service_class_id - Service class ID monitor element

Unique ID of service subclass. For a unit of work, this ID represents the service subclass ID of the workload with which the connection issuing the unit of work is associated.

Table 1149. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	Always collected
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected

Table 1149. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics	Always collected
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 1150. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Locking	-	-
Unit of work	-	-
Statistics	event_histogrambin	-
Statistics	event_scstats	-

Usage

The value of this element matches a value from column SERVICECLASSID of view SYSCAT.SERVICECLASSES. Use this element to look up the service subclass name, or link information about a service subclass from different sources. For example, join service class statistics with histogram bin records.

The value of this element is 0 when the following conditions are met:

- The element is reported in an event_histogrambin logical data group.
- The histogram data is collected for an object that is not a service class.

service_level - Service Level

This is the current corrective service level of the DB2 instance.

Table 1151. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

service_subclass_name - Service subclass name monitor element

The name of a service subclass.

Table 1152. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	Always collected

Table 1152. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected

Table 1153. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Locking	-	-
Unit of work	-	-
Activities	event_activity	-
Statistics	event_scstats	-
Statistics	event_qstats	-

Usage

Use this element in conjunction with other activity elements for analysis of the behavior of an activity or with other statistics elements for analysis of a service class or threshold queue.

service_superclass_name - Service superclass name monitor element

The name of a service superclass.

Table 1154. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	Always collected
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected

Table 1155. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	-	-
Activities	event_activity	-
Statistics	event_scstats	-
Statistics	event_qstats	-

Usage

Use this element in conjunction with other activity elements for analysis of the behavior of an activity or with other statistics elements for analysis of a service class or threshold queue.

session_auth_id - Session authorization ID monitor element

The current authorization ID for the session being used by this application. For monitoring workload management activities, this monitor element describes the session authorization ID under which the activity was injected into the system.

Table 1156. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	Always collected
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected

Table 1157. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
Lock	appl_lock_list	Basic

Table 1158. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-
Activities	event_activity	-
Threshold violations	event_activity event_thresholdviolations	-

Usage

You can use this element to determine what authorization ID is being used to prepare SQL statements, execute SQL statements, or both. This monitor element does not report any session authorization ID values set within executing stored procedures.

shr_workspace_num_overflows - Shared Workspace Overflows

The number of times that shared workspaces overflowed the bounds of their allocated memory.

Note: This monitor element has been deprecated. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

Table 1159. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 1160. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage Use this element with shr_workspace_size_top to determine whether the size of the Shared Workspaces need to be increased to avoid overflowing. Overflows of Shared Workspaces may cause performance degradation as well as out of memory errors from the other heaps allocated out of application shared memory.

At the database level, the element reported will be from the same shared workspace as that which was reported as having the Maximum Shared Workspace Size. At the application level, it is the number of overflows for the workspace used by the current application.

shr_workspace_section_inserts - Shared Workspace Section Inserts

Number of inserts of SQL sections by applications into shared workspaces.

Note: This monitor element has been deprecated. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

Table 1161. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 1162. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage The working copy of executable sections are stored in shared workspaces. This counter indicates when a copy was not available and had to be inserted.

At the database level, it is the cumulative total of all inserts for every application across all shared workspaces in the database. At the application level, it is the cumulative total of all inserts for all sections in the shared workspace for this application.

shr_workspace_section_lookups - Shared Workspace Section Lookups

Lookups of SQL sections by applications in shared workspaces.

Note: This monitor element has been deprecated. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

Table 1163. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 1164. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage Each application has access to a shared workspace where the working copy of executable sections are kept.

This counter indicates how many times shared workspaces were accessed in order to locate a specific section for an application. At the database level, it is the cumulative total of all lookups for every application across all Shared Workspaces in the database. At the application level, it is the cumulative total of all lookups for all sections in the shared workspace for this application.

You can use this element in conjunction with Shared Workspace Section Inserts to tune the size of shared workspaces. The size of the shared workspace is controlled by the app_ctl_heap_sz configuration parameter.

shr_workspace_size_top - Maximum Shared Workspace Size

The largest size reached by shared workspaces.

Note: This monitor element has been deprecated. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

Table 1165. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

Table 1166. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage This element indicates the maximum number of bytes the shared workspaces required for the workload run against the database since it was activated. At the database level, it is the maximum size reached by all of the shared workspaces. At the application level, it is the maximum size of the shared workspace used by the current application.

If a shared workspace overflowed, then this element contains the largest size reached by that shared workspace during the overflow. Check Shared Workspace Overflows to determine if such a condition occurred.

When the shared workspace overflows, memory is temporarily borrowed from other entities in application shared memory. This can result in memory shortage errors from these entities or possibly performance degradation. You can reduce the chance of overflow by increasing APP_CTL_HEAP_SZ.

smallest_log_avail_node - Node with Least Available Log Space

This element is only returned for global snapshots and indicates the node with the least amount (in bytes) of available log space.

Element identifier
smallest_log_avail_node

Element type
information

Table 1167. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Usage Use this element, in conjunction with appl_id_oldest_xact, to ensure that adequate log space is available for the database. In a global snapshot, appl_id_oldest_xact, total_log_used, and total_log_available correspond to the values on this node.

sort_heap_allocated - Total Sort Heap Allocated

The total number of allocated pages of sort heap space for all sorts at the level chosen and at the time the snapshot was taken.

Table 1168. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic
Database	dbase	Basic

Usage The amount of memory allocated for each sort may be some or all of the available sort heap size. Sort heap size is the amount of memory available for each sort as defined in the *sortheap* database configuration parameter.

It is possible for a single application to have concurrent sorts active. For example, in some cases a SELECT statement with a subquery can cause concurrent sorts.

Information may be collected at two levels:

- At the database manager level, it represents the sum of sort heap space allocated for all sorts in all active databases in the database manager
- At the database level, it represents the sum of the sort heap space allocated for all sorts in a database.

Normal memory estimates do not include sort heap space. If excessive sorting is occurring, the extra memory used for the sort heap should be added to the base memory requirements for running the database manager. Generally, the larger the sort heap, the more efficient the sort. Appropriate use of indexes can reduce the amount of sorting required.

You may use the information returned at the database manager level to help you tune the *sheapthres* configuration parameter. If the element value is greater than or equal to *sheapthres*, it means that the sorts are not getting the full sort heap as defined by the *sortheap* parameter.

sort_heap_top - Sort private heap high watermark

The private sort memory high watermark, in 4 KB pages, across the database manager.

Table 1169. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

Usage This element can be used to determine if the SHEAPTHRES configuration parameter is set to an optimal value. For example, if this watermark approaches or exceeds SHEAPTHRES, it is likely that SHEAPTHRES should be increased. This is because private sorts are given less memory whenever SHEAPTHRES is exceeded, and this can adversely affect system performance.

sort_overflows - Sort overflows monitor element

The total number of sorts that ran out of sort heap and may have required disk space for temporary storage.

Table 1170. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 1171. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic
Application	stmt	Basic
Dynamic SQL	dynsql	Basic

For snapshot monitoring, this counter can be reset.

Table 1172. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Database	event_db	-
Connection	event_conn	-
Statements	event_stmt	-
Activities	event_activity	Statement, Sort
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

At a database or application level, use this element in conjunction with **total_sorts** to calculate the percentage of sorts that had to overflow to disk. If this percentage is high, you may want adjust the database configuration by increasing the value of **sortheap**.

At a statement level, use this element to identify statements that require large sorts. These statements may benefit from additional tuning to reduce the amount of sorting required.

When a sort overflows, additional overhead will be incurred because the sort will require a merge phase and can potentially require more I/O, if data needs to be written to disk.

This element provides information for one statement, one application, or all applications accessing one database.

sort_shrheap_allocated - Sort Share Heap Currently Allocated

Total amount of shared sort memory allocated in the database.

Table 1173. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Usage This element can be used to assess the threshold for shared sort memory. If

this value is frequently much higher or lower than the current shared sort memory threshold, it is likely that the threshold should be adjusted.

Note: The "shared sort memory threshold" is determined by the value of the SHEAPTHRES database manager configuration parameter if the SHEAPTHRES_SHR database configuration parameter is 0. Otherwise, it is determined by the value of SHEAPTHRES_SHR.

sort_shrheap_top - Sort share heap high watermark

Database-wide shared sort memory high watermark in 4 KB pages.

Table 1174. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Usage This element can be used to assess whether or not SHEAPTHRES (or SHEAPTHRES_SHR) is set to an optimal value. For example, if this high watermark is persistently much lower than the shared sort memory threshold, it is likely that this threshold needs to be decreased, thus freeing memory for other database functions. Conversely, if this high watermark begins to approach the shared sort memory threshold, then this might indicate that this threshold needs to be increased. This is important because the shared sort memory threshold is a hard limit. When the total amount of sort memory reaches this threshold, no more shared sorts can be initiated.

This element, along with the high watermark for private sort memory, can also help users determine if the threshold for shared and private sorts need to be set independently of each other. Normally, if the SHEAPTHRES_SHR database configuration option has a value of 0, then the shared sort memory threshold is determined by the value of the SHEAPTHRES database manager configuration option. However, if there is a large discrepancy between the private and shared sort memory high watermarks, this might be an indication that the user needs to override SHEAPTHRES and set SHEAPTHRES_SHR to a more appropriate value that is based on the shared sort memory high watermark.

Note: This element reports the high watermark of sort reservation requests granted by the sort memory controller. Requests that are granted do not always result in a similar level of memory allocation, since they only permit consumers of sort heap to allocate memory as necessary, up to the granted amount, during the processing of an SQL request. It is normal for there to be a discrepancy between the value for this element and the high water mark of the shared sort memory pool (pool_watermark).

source_service_class_id - Source service class ID monitor element

The ID of the service subclass from which an activity was remapped when the threshold violation record to which this element belongs was generated. This element has a value of zero when the threshold action is anything other than a REMAP ACTIVITY action.

Table 1175. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Threshold violations	event_thresholdviolations	-

Usage

Use this element to trace the path of an activity through the service classes to which it was remapped. It can also be used to compute aggregates of how many activities were mapped out of a given service subclass.

sp_rows_selected - Rows Returned by Stored Procedures

This element contains the number of rows sent from the data source to the federated server at the start of the federated server instance, or the last reset of the database monitor counters as a result of stored procedure operations for this application.

Table 1176. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

Usage This element has several uses. You can use it to compute the average number of rows sent to the federated server from the data source, per stored procedure, with the following formula:

```
   rows per stored procedure
 = rows returned
 / # of stored procedures invoked
```

You can also compute the average time to return a row to the federated server from the data source for this application:

```
average time = aggregate stored proc. response time / rows returned
```

sql_chains - Number of SQL Chains Attempted

Represents the number of SQL statements taking *n* data transmissions between the DB2 Connect gateway and the host during statement processing. The range *n* is specified by the *num_transmissions_group* element.

Table 1177. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Data Transmission	stmt_transmissions	Basic

For snapshot monitoring, this counter can be reset.

For example, if chaining is on, and if PREP and OPEN statements are chained together and the chain takes a total of two transmissions, *sql_chains* is reported as "1" and *sql_stmts* is reported as "2".

If chaining is off, then the *sql_chains* count equals the *sql_stmts* count.

Usage Use this element to get statistics on how many statements used 2, 3, 4 (and so on) data transmissions during their processing. (At least two data transmissions are necessary to process a statement: a send and a receive.) These statistics can give you a better idea of the database or application activity and network traffic at the database or application levels.

Note: The *sql_stmts* monitor element represents the number of attempts made to send an SQL statement to the server. At the transmission level, all statements within the same cursor count as a single SQL statement.

sql_req_id - Request Identifier for SQL Statement

The request identifier for an operation in an SQL statement.

Table 1178. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_stmt	-

Usage This identifier increments with each successive SQL operation processed by the database manager since the first application has connected to the database. Its value is unique across the database and uniquely identifies a statement operation.

sql_reqs_since_commit - SQL Requests Since Last Commit

Number of SQL requests that have been submitted since the last commit.

Table 1179. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

Usage You can use this element to monitor the progress of a transaction.

sql_stmts - Number of SQL Statements Attempted

For data transmission snapshots, this element represents the number of SQL statements taking *n* data transmissions between the DB2 Connect gateway and the host during statement processing. The range *n* is specified by the *num_transmissions_group* element.

Table 1180. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Basic
DCS Application	dcs_appl	Basic
Data Transmission	stmt_transmissions	Basic

For snapshot monitoring, this counter can be reset.

For DCS DATABASE snapshots, this statement count is the number of statements since the database was activated.

For DCS APPLICATION snapshots, this statement count is the number of statements since the connection to the database was established by this application.

Usage Use this element to measure database activity at the database or application level. To calculate the SQL statement throughput for a given period, you can divide this element by the elapsed time between two snapshots.

For the data transmission level: Use this element to get statistics on how many statements used 2, 3, 4 (and so on) data transmissions during their processing. (At least 2 data transmissions are necessary to process a statement: a send and a receive.) These statistics can give you a better idea of the database or application activity and network traffic at the database or application levels.

Note:

1. The *sql_stmts* monitor element represents the number of attempts made to send an SQL statement to the server:
   - At the application level and database level, each SQL statement within a cursor is counted separately.
   - At the transmission level, all statements within the same cursor count as a single SQL statement.

sqlca - SQL Communications Area (SQLCA)

The SQLCA data structure that was returned to the application at statement completion.

Table 1181. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_stmt	-
Activities	event_activity	-

Usage

The SQLCA data structure can be used to determined if the statement completed successfully. For information about the content of the SQLCA, see "SQLCA (SQL communications area)" in *SQL Reference, Volume 1* or "SQLCA data structure" in *Administrative API Reference* .

sqlrowsread_threshold_id - SQL rows read threshold ID monitor element

The ID of the SQLROWSREAD threshold that was applied to the activity.

Table 1182. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand which SQLROWSREAD threshold, if any, was applied to the activity.

sqlrowsread_threshold_value - SQL rows read threshold value monitor element

The upper bound of the SQLROWSREAD threshold that was applied to the activity.

Table 1183. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand the value of the SQLROWSREAD threshold applied to the activity, if any.

sqlrowsread_threshold_violated - SQL rows read threshold violated monitor element

This monitor element returns 'Yes' to indicate that the activity violated the SQLROWSREAD threshold. 'No' indicates that the activity has not yet violated the threshold.

Table 1184. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to determine if the activity violated the SQLROWSREAD threshold that was applied to the activity.

sqlrowsreadinsc_threshold_id - SQL rows read in service class threshold ID monitor element

The ID of the SQLROWSREADINSC threshold that was applied to the activity.

Table 1185. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand which SQLROWSREADINSC threshold, if any, was applied to the activity.

sqlrowsreadinsc_threshold_value - SQL rows read in service class threshold value monitor element

The upper bound of the SQLROWSREADINSC threshold that was applied to the activity.

Table 1186. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand the value of the SQLROWSREADINSC threshold applied to the activity, if any.

sqlrowsreadinsc_threshold_violated - SQL rows read in service class threshold violated monitor element

This monitor element returns 'Yes' to indicate that the activity violated the SQLROWSREADINSC threshold. 'No' indicates that the activity has not yet violated the threshold.

Table 1187. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to determine if the activity violated the SQLROWSREADINSC threshold that was applied to the activity.

sqlrowsreturned_threshold_id - SQL rows read returned threshold ID monitor element

The ID of the SQLROWSRETURNED threshold that was applied to the activity.

Table 1188. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand which SQLROWSRETURNED threshold, if any, was applied to the activity.

sqlrowsreturned_threshold_value - SQL rows read returned threshold value monitor element

The upper bound of the SQLROWSRETURNED threshold that was applied to the activity.

Table 1189. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand the value of the SQLROWSRETURNED threshold applied to the activity, if any.

sqlrowsreturned_threshold_violated - SQL rows read returned threshold violated monitor element

This monitor element returns 'Yes' to indicate that the activity violated the SQLROWSRETURNED threshold. 'No' indicates that the activity has not yet violated the threshold.

Table 1190. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to determine if the activity violated the SQLROWSRETURNED threshold that was applied to the activity.

sqltempspace_threshold_id - SQL temporary space threshold ID monitor element

The ID of the SQLTEMPSPACE threshold that was applied to the activity.

Table 1191. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand which SQLTEMPSPACE threshold, if any, was applied to the activity.

sqltempspace_threshold_value - SQL temporary space threshold value monitor element

The upper bound of the SQLTEMPSPACE threshold that was applied to the activity.

Table 1192. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to understand the value of the SQLTEMPSPACE threshold applied to the activity, if any.

sqltempspace_threshold_violated - SQL temporary space threshold violated monitor element

This monitor element returns 'Yes' to indicate that the activity violated the SQLTEMPSPACE threshold. 'No' indicates that the activity has not yet violated the threshold.

Table 1193. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Usage

Use this element to determine if the activity violated the SQLTEMPSPACE threshold that was applied to the activity.

ss_exec_time - Subsection Execution Elapsed Time

The time in seconds that it took a subsection to execute.

Table 1194. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

Table 1195. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_subsection	-

Usage Allows you to track the progress of a subsection.

ss_node_number - Subsection Node Number

Node where the subsection was executed.

Table 1196. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

Table 1197. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_subsection	-

Usage Use to correlate each subsection with the database partition where it was executed.

ss_number - Subsection number monitor element

Identifies the subsection associated with the returned information.

Table 1198. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_APPL_LOCKWAIT table function - get information about locks for which an application is waiting	

Table 1199. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

Table 1200. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_subsection	-

Usage

This number relates to the subsection number in the access plan that can be obtained with **db2expln** command.

ss_status - Subsection Status

The current status of an executing subsection.

Table 1201. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

Usage The current status values can be:

- executing (SQLM_SSEXEC in sqlmon.h)
- waiting for a lock

- waiting to receive data on a table queue
- waiting to send data on a table queue

ss_sys_cpu_time - System CPU Time used by Subsection

The total system CPU time (in seconds and microseconds) used by the currently executing statement subsection. For event monitors that write to tables, the value of this element is given in microseconds by using the BIGINT data type.

Element identifier
ss_sys_cpu_time

Element type
time

Table 1202. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Timestamp

Table 1203. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_subsection	Timestamp

Usage This element along with the other related CPU-time elements can help you understand the level of activity within an application, and can help you identify applications that could benefit from additional tuning.

System CPU represents the time spent in system calls. User CPU represents time spent executing database manager code.

ss_usr_cpu_time - User CPU Time used by Subsection

The total user CPU time (in seconds and microseconds) used by the currently executing statement subsection. For event monitors that write to tables, the value of this element is given in microseconds by using the BIGINT data type.

Element identifier
ss_usr_cpu_time

Element type
time

Table 1204. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Timestamp

Table 1205. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_subsection	Timestamp

Usage This element along with the other related CPU-time elements can help you understand the level of activity within an application, and can help you identify applications that could benefit from additional tuning.

System CPU represents the time spent in system calls. User CPU represents time spent executing database manager code.

start_time - Event Start Time

The date and time of unit of work start, statement start, or deadlock detection. This element, in the event_start API structure indicates the start of the event monitor.

Table 1206. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_start	Timestamp
Statements	event_stmt	Timestamp
Deadlocks	event_deadlock	Timestamp
Deadlocks	event_dlconn	Timestamp
Deadlocks with Details	event_detailed_dlconn	Timestamp

Usage You can use this element to correlate the deadlock connection records to the deadlock event record, and in conjunction with *stop_time* to calculate the elapsed statement or transaction execution time.

Note: When the Timestamp switch is OFF, this element reports "0".

static_sql_stmts - Static SQL Statements Attempted

The number of static SQL statements that were attempted.

Table 1207. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 1208. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage You can use this element to calculate the total number of successful SQL statements at the database or application level:

```
  dynamic_sql_stmts
+ static_sql_stmts
- failed_sql_stmts
= throughput during monitoring period
```

statistics_timestamp - Statistics timestamp monitor element

The time at which this statistics record was generated.

Table 1209. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wlstats	-
Statistics	event_wcstats	-
Statistics	event_qstats	-
Statistics	event_histogrambin	-

Usage

Use this element to determine when this statistics record was generated.

Use this element along with the **last_wlm_reset** element to identify the time interval over which the statistics in this statistics record were generated.

This monitor element can also be used to group together all statistics records that were generated for the same collection interval.

stats_cache_size – Size of statistics cache monitor element

The current size of the statistics cache, in bytes, which is used in a catalog partition to cache statistics information generated by real-time statistics gathering.

Note: Since the statistics cache resides in the catalog partition, only the snapshot taken at the catalog partition will report the statistics cache size. Snapshots taken at other partitions will report the value of zero instead. When taking a global snapshot, the values reported by all the database partitions are aggregated together.

Table 1210. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	-

Table 1211. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage

Use this element to determine the size of the current statistics cache. This value changes frequently. In order to evaluate system usage, take the snapshot at specific intervals over an extended period of time. Use this element to adjust the value of the **catalogcache_sz** configuration parameter.

stats_fabricate_time – Total time spent on statistics fabrication activities monitor element

The total time spent on statistics fabrications by real-time statistics gathering, in milliseconds. Statistics fabrication is the statistics collection activity needed to generate statistics during query compilation. If this monitor element is collected at the database level, it represents the total time spent on real-time statistics gathering activities for all the applications running on the database. If it is collected at the statement level, it represents the time spent on the latest real-time statistics gathering activities for the statement. The times reported by all the database partitions are aggregated together.

Table 1212. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Statement
Dynamic SQL	dynsql	Statement

For snapshot monitoring, this element can be reset.

Table 1213. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Statement	event_stmt	-

Usage

Use this element along with **stats_fabrications** to evaluate the performance impact of real-time statistics gathering at the database level. For snapshot monitor for dynamic SQL, you can use this element along with **total_exec_time** and **num_executions** to evaluate the impact of statistics fabrications. For the statement event monitor, you can combine this element with **stmt_start** and **stmt_stop** for further evaluation of real-time statistics gathering impact.

stats_fabrications – Total number of statistics fabrications monitor elements

The total number of statistics fabrications performed by real-time statistics during query compilation for all the database applications. Rather than obtaining statistics by scanning data stored in a table or an index, statistics are fabricated based on metadata maintained by the index and data manager. Values reported by all the database partitions are aggregated together.

Table 1214. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Statement

For snapshot monitoring, this counter can be reset.

Table 1215. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage

Use this element to determine the frequency of statistics fabrications in the database. This value changes frequently. In order to get a better overview of the system usage, take the snapshot at specific intervals over an extended period of time. When used in conjunction with **stats_fabricate_time**, this element can help you evaluate the impact of statistics fabrications.

status_change_time - Application Status Change Time

The date and time the application entered its current status.

Element identifier
status_change_time

Element type
timestamp

Table 1216. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_id_info	Unit of Work, Timestamp
Lock	appl_lock_list	Unit of Work, Timestamp
DCS Application	dcs_appl_info	Unit of Work, Timestamp

Usage This element allows you to determine how long an application has been in its current status. If it has been in the same status for a long period of time, this may indicate that it has a problem.

stmt_elapsed_time - Most Recent Statement Elapsed Time

The elapsed execution time of the most recently completed statement.

Table 1217. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement, Timestamp
DCS Statement	dcs_stmt	Statement, Timestamp

Usage

Use this element as an indicator of the time it takes for a statement to complete.

This element is composed of two subelements that report time spent as seconds and microseconds (one millionth of a second). The names of the subelements can be derived by adding "_s" and "_ms" to the name of this monitor element. To retrieve the total time spent for this monitor element, the values of the two subelements must be added together. For example, if the "_s" subelement value is 3 and the "_ms" subelement value is 20, then the total time spent for the monitor element is 3.00002 seconds.

stmt_exec_time - Statement execution time monitor element

The total time spent executing this statement by all agents on this member. The value is given in milliseconds.

Table 1218. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 1219. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE

stmt_first_use_time - Statement first use timestamp monitor element

This element shows the first time the statement entry was processed. For cursor operations, **stmt_first_use_time** shows when the cursor was opened. At application coordination nodes, this value reflects the application requests; at non-coordinator nodes, this value reflects when requests were received from the originating node.

Table 1220. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values [1]	event_stmt_history	timestamp
Deadlocks with Details History [1]	event_stmt_history	timestamp
Activities	event_activitystmt	timestamp

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

Use this element in conjunction with other statement history entries to see the sequence of SQL statements that caused the deadlock.

stmt_history_id - Statement history identifier

This numerical element shows the position in which the statement was run within the unit of work indicated by the sequence_no element, relative to other statement history elements. The earliest statement run in the unit of work will have the lowest value. If the same statement is run twice in the same unit of work, two different occurrences of the statement will be shown with two different stmt_history_id values.

Table 1221. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks with Details History Values	event_stmt_history	-
Deadlocks with Details History Values	event_data_value	-
Deadlocks with Details History	event_stmt_history	-

Usage You can use this information to see the sequence of SQL statements that caused the deadlock.

inact_stmthist_sz - Statement history list size

When a detailed deadlock event monitor with history is running, this element reports the number of bytes being used from the database monitor heap (MON_HEAP_SZ) to keep track of the statement history list entries.

Table 1222. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
application	appl	-
database	db	-

Usage You can use this element when tuning the database monitor heap.

stmt_invocation_id - Statement invocation identifier monitor element

An identifier that distinguishes one invocation of a routine from others at the same nesting level within a unit of work. It is unique within a unit of work for a specific nesting level.

Table 1223. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 1224. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activitystmt	-
Locking	-	-
Deadlocks with Details History Values[1]	event_stmt_history	-
Deadlocks with Details History[1]	event_stmt_history	-
Unit of work	Reported in the package list.	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

You can use this element to uniquely identify the invocation in which a particular SQL statement has been executed. You can also use this element in conjunction with other statement history entries to see the sequence of SQL statements that caused the deadlock.

stmt_isolation - Statement isolation

This element shows the isolation value in effect for the statement while it was being run.

Table 1225. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks with Details History Values	event_stmt_history	-
Deadlocks with Details History	event_stmt_history	-
Activities	event_activitystmt	-

The possible isolation level values are:

- SQLM_ISOLATION_LEVEL_NONE 0 (no isolation level specified)
- SQLM_ISOLATION_LEVEL_UR 1 (uncommitted read)
- SQLM_ISOLATION_LEVEL_CS 2 (cursor stability)
- SQLM_ISOLATION_LEVEL_RS 3 (read stability)
- SQLM_ISOLATION_LEVEL_RR 4 (repeatable read)

Usage You can use this element in conjunction with other statement history entries to understand the cause of the deadlock and the execution behavior of a particular SQL statement.

stmt_last_use_time - Statement last use timestamp monitor element

This element shows the last time the statement entry was processed. For cursor operations, **stmt_last_use_time** shows the time of the last action on the cursor where that action could be an open, fetch, or close. At application coordination nodes, this value reflects the application requests; at non-coordinator nodes, this value reflects when requests were received from the originating node.

Table 1226. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values [1]	event_stmt_history	timestamp
Deadlocks with Details History [1]	event_stmt_history	timestamp
Activities	event_activitystmt	timestamp

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

Use this element in conjunction with other statement history entries to see the sequence of SQL statements that caused the deadlock.

stmt_lock_timeout - Statement lock timeout monitor element

This element shows the lock timeout value in effect for the statement while it was being run.

Table 1227. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values[1]	event_stmt_history	-
Deadlocks with Details History[1]	event_stmt_history	-
Activities	event_activitystmt	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

You can use this element in conjunction with other statement history entries to understand the cause of the deadlock and the execution behavior of a particular SQL statement.

stmt_nest_level - Statement nesting level monitor element

This element shows the level of nesting or recursion in effect when the statement was being run; each level of nesting corresponds to nested or recursive invocation of a stored procedure or user-defined function (UDF).

Table 1228. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 1229. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values[1]	event_stmt_history	-
Deadlocks with Details History[1]	event_stmt_history	-
Activities	event_activitystmt	-
Unit of work	Reported in the package list.	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

You can use this element, along with **stmt_invocation_id** monitor element, to uniquely identify the invocation in which a particular SQL statement has been executed. You can also use this element in conjunction with other statement history entries to see the sequence of SQL statements that caused the deadlock.

stmt_node_number - Statement Node

Node where the statement was executed.

Table 1230. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement

Usage Used to correlate each statement with the node where it was executed.

stmt_operation/operation - Statement operation monitor element

The statement operation currently being processed or most recently processed (if none currently running).

Table 1231. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement

Table 1231. Snapshot Monitoring Information (continued)

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Statement	dcs_stmt	Statement

Table 1232. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details[1]	event_detailed_dlconn	-
Statements	event_stmt	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

You can use this element to determine the operation that is executing or recently finished.

It can be one of the following.

For SQL operations:
- SELECT
- PREPARE
- EXECUTE
- EXECUTE IMMEDIATE
- OPEN
- FETCH
- CLOSE
- DESCRIBE
- STATIC COMMIT
- STATIC ROLLBACK
- FREE LOCATOR
- PREP_COMMIT
- CALL
- PREP_OPEN
- PREP_EXEC
- COMPILE
- DROP PACKAGE

For non-SQL operations:
- RUN STATISTICS
- REORG
- REBIND
- REDISTRIBUTE
- GET TABLE AUTHORIZATION

- GET ADMINISTRATIVE AUTHORIZATION

Note: API users should refer to the `sqlmon.h` header file containing definitions of database system monitor constants.

stmt_pkgcache_id - Statement package cache identifier monitor element

This element shows the internal package cache identifier (ID) for a dynamic SQL statement.

Table 1233. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	Always collected

Table 1234. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Dynamic SQL	dynsql	Basic

Table 1235. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values[1]	event_stmt_history	-
Deadlocks with Details History[1]	event_stmt_history	-
Activities	event_activitystmt	-
Package cache	-	COLLECT BASE DATA

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

In a multi-partitioned environment, each partition has a unique statement ID for a cached statement. A given statement may not have the same ID across partitions.

In a global dynamic SQL snapshot, only the first statement ID is returned.

stmt_query_id - Statement query identifier monitor element

This element shows the internal query identifier (ID) given to any SQL statement used as a cursor.

Table 1236. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values[1]	event_stmt_history	-
Deadlocks with Details History[1]	event_stmt_history	-
Activities	event_activitystmt	-

Usage

You can use this element, along with the **stmt_nest_level** monitor element, to uniquely identify an invocation of a particular SQL statement. You can also use this element in conjunction with other statement history entries to understand the cause of the deadlock.

stmt_sorts - Statement Sorts

The total number of times that a set of data was sorted in order to process the stmt_operation.

Table 1237. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Statement
Application	stmt	Statement
Dynamic SQL	dynsql	Statement

Usage You can use this element to help identify the need for an index, since indexes can reduce the need for sorting of data. Using the related elements in the above table you can identify the SQL statement for which this element is providing sort information, and then analyze this statement to determine index candidates by looking at columns that are being sorted (for example, columns used in ORDER BY and GROUP BY clauses and join columns). See **explain** in the *Administration Guide* for information on checking whether your indexes are used to optimize sort performance.

This count includes sorts of temporary tables that were generated internally by the database manager to execute the statement. The number of sorts is associated with the first FETCH operation of the SQL statement. This information is returned to you when the operation for the statement is the first FETCH. You should note that for blocked cursors several fetches may be performed when the cursor is opened. In these cases it can be difficult to use the snapshot monitor to obtain the number of sorts, since a snapshot would need to be taken while DB2 was internally issuing the first FETCH.

A more reliable way to determine the number of sorts performed when using a blocked cursor would be with an event monitor declared for statements. The total_sorts counter, in the statement event for the CLOSE

cursor, contains the total number of sorts that were performed while executing the statement for which the cursor was defined.

stmt_source_id - Statement source identifier

This element shows the internal identifier (ID) given to the source of the SQL statement that was run.

Table 1238. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values[1]	event_stmt_history	-
Deadlocks with Details History[1]	event_stmt_history	-
Activities	event_activitystmt	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

You can use this element, along with **appl_id** monitor element, to uniquely identify the origin of a request to run a particular SQL statement. You can also use this element in conjunction with other statement history entries to understand the cause of the deadlock.

stmt_start - Statement Operation Start Timestamp

The date and time when the stmt_operation started executing.

Element identifier
stmt_start

Element type
timestamp

Table 1239. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement, Timestamp
DCS Statement	dcs_stmt	Statement, Timestamp

Usage You can use this element with stmt_stop to calculate the elapsed statement operation execution time.

stmt_stop - Statement Operation Stop Timestamp

The date and time when the stmt_operation stopped executing.

Element identifier
stmt_stop

Element type
Timestamp

Table 1240. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement, Timestamp
DCS Statement	dcs_stmt	Statement, Timestamp

Usage You can use this element with stmt_start to calculate the elapsed statement operation execution time.

stmt_sys_cpu_time - System CPU Time used by Statement

The total *system* CPU time (in seconds and microseconds) used by the currently executing statement.

Element identifier
stmt_sys_cpu_time

Element type
time

Table 1241. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Statement, Timestamp
Application	stmt	Statement, Timestamp

Usage This element along with the other related CPU-time elements can help you understand the level of activity within an application, and can help you identify applications that could benefit from additional tuning.

This counter includes time spent on both SQL and non-SQL statements, as well as any unfenced user defined functions (UDF) or stored procedures executed by the application.

System CPU represents the time spent in system calls. User CPU represents time spent executing database manager code.

Note: If this information is not available for your operating system, this element will be set to 0.

stmt_text - SQL statement text monitor element

The text of the SQL statement.

Table 1242. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected

Table 1242. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	Always collected

Table 1243. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement
Dynamic SQL	dynsql	Basic
DCS Statement	dcs_stmt	Statement

Table 1244. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details[1]	event_detailed_dlconn	-
Deadlocks with Details History[1]	event_stmt_history	-
Statements	event_stmt	-
Activities	event_activitystmt	-
Package cache	-	COLLECT BASE DATA

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

For application snapshots, this statement text helps you identify what the application was executing when the snapshot was taken, or most recently processed if no statement was being processed right at the time the snapshot was taken.

The information returned by this element is taken from the SQL statement cache and it might not be available if the cache has overflowed. The only guaranteed way to capture the SQL text of a statement is to use an event monitor for statements.

For dynamic SQL statements, this element identifies the SQL text associated with a package.

For statement event monitors, this element is returned only for dynamic statements. If a statement event monitor record cannot fit into the size of the buffer specified by the BUFFERSIZE option of a statement event monitor, the value of the **stmt_text** monitor may be truncated so that the record can fit.

For the EVENT_STMT_HISTORY event monitor, this element is returned only for dynamic statements. For remaining event monitors, **stmt_text** is returned for dynamic and static statements only if it is available in the SQL statement cache.

For information on how to query the system catalog tables to obtain static SQL statement text that is not provided due to performance considerations, see the **section_number** monitor element.

stmt_type - Statement type monitor element

The type of statement processed.

Table 1245. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement

Table 1246. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details[1]	event_detailed_dlconn	-
Statements	event_stmt	-
Activities	event_activitystmt	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

You can use this element to determine the type of statement that is executing. It can be one of the following:

- A static SQL statement
- A dynamic SQL statement
- An operation other than an SQL statement; for example, a bind or pre-compile operation.

For the snapshot monitor, this element describes the statement that is currently being processed or was most recently processed.

Note: API users should refer to the `sqlmon.h` header file containing definitions of database system monitor constants.

stmt_type_id - Statement type identifier monitor element

Statement type identifier.

Table 1247. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	Always collected

Table 1248. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Package cache	-	COLLECT BASE DATA

Usage

The **stmt_type_id** monitor element has the following possible values:

- Statement not prepared
- DDL, (not Set Constraints)
- DDL, Set Constraints
- DML, Select
- DML, Insert/Update/Delete
- Authorization
- DML, Select (blockable)
- DML, Lock Table
- DML, Commit/Rollback
- Set environment
- DDL, Savepoint
- DDL, (declared user temp)
- Passthru support
- CALL
- Free locator
- DML, Select with IUD
- DML, Select with IUD (blockable)
- Top-level SET, no SQL
- Top-level SET, reads SQL
- DDL, (issues internal commit)
- Top-level SET, modifies SQL
- Unknown

stmt_usr_cpu_time - User CPU Time used by Statement

The total *user* CPU time (in seconds and microseconds) used by the currently executing statement.

Element identifier
stmt_usr_cpu_time

Element type
time

Table 1249. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Statement, Timestamp
Application	stmt	Statement, Timestamp

Usage This element along with the other related CPU-time elements can help you

understand the level of activity within an application, and can help you identify applications that could benefit from additional tuning.

This counter includes time spent on both SQL and non-SQL statements, as well as any unfenced user-defined functions (UDFs) or stored procedures executed by the application.

System CPU represents the time spent in system calls. User CPU represents time spent executing database manager code.

Note: If this information is not available for your operating system, this element will be set to 0.

stmt_value_data - Value data

This element contains a string representation of a data value to an SQL statement. LOB, LONG, and structured type parameters appear as empty strings. Date, time, and timestamp fields are recorded in ISO format.

Table 1250. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values[1]	stmt_value_data	-
Activities	event_activityvals	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

You can use this element in conjunction with other statement history entries to understand the cause of the deadlock.

stmt_value_index - Value index

This element represents the position of the input parameter marker or host variable used in the SQL statement.

Table 1251. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values[1]	stmt_value_data	-
Activities	event_activityvals	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

You can use this element in conjunction with other statement history entries to understand the cause of the deadlock.

stmt_value_isnull - Value has null value monitor element

This element shows whether a data value that is associated with an SQL statement is the NULL value; whether an extended indicator has been used to specify the default value; or that this statement value is unassigned.

Possible values are:

- 0 or "no" if the value is not NULL
- 1 or "yes" if the value is NULL
- 2 or "default" if the extended indicator value of default (-5) was specified for this statement value
- 3 or "unassigned" if the extended indicator value of unassigned (-7) was specified for this statement value

Table 1252. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values[1]	stmt_value_isnull	-
Activities	event_activityvals	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

You can use this element in conjunction with other statement history entries to understand the cause of the deadlock.

stmt_value_isreopt - Variable used for statement reoptimization monitor element

This element shows whether the provided value was a value used during statement reoptimization. It returns a value of “True” if the statement was reoptimized (for example, due to the setting of the REOPT bind option) and if the value was used as input to the SQL compiler during this reoptimization.

Table 1253. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values[1]	event_data_value	-
Activities	event_activityvals	-

1 This option has been deprecated. Its use is no longer recommended and

might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

You can use this element in conjunction with the provided compilation environment to allow for full analysis of the SQL compiler's treatment of the SQL statement.

stmt_value_type - Value type monitor element

This element contains a string representation of the type of a data value associated with an SQL statement.

Table 1254. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values[1]	stmt_value_type	-
Activities	event_activityvals	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

You can use this element in conjunction with other statement history entries to understand the cause of the deadlock.

sto_path_free_sz - Automatic storage path free space monitor element

This element shows the amount of free space (in bytes) available on a file system pointed to by a storage path. If multiple storage paths point to the same file system, the free size is not divided among them.

Table 1255. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	db_sto_path_info	Buffer Pool

Usage

You can use this element together with the following elements to gather per-node data on space utilization for the database:

- **db_storage_path**
- **fs_used_size**
- **fs_total_size**
- **fs_id**

stop_time - Event Stop Time

The date and time when the statement stopped executing.

Table 1256. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_stmt	Timestamp

Usage You can use this element with *start_time* to calculate the elapsed statement execution time.

For a FETCH statement event, this is the time of the last successful fetch.

Note: When the Timestamp switch is OFF, this element reports "0".

stored_proc_time - Stored Procedure Time

This element contains the aggregate amount of time, in milliseconds, that it has taken this data source to respond to stored procedure statements from all applications or a single application running on this federated server instance from the start of the federated server instance or the last reset of the database monitor counters.

Table 1257. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Timestamp
Application	appl_remote	Timestamp

For snapshot monitoring, this counter can be reset.

The response time is measured as the difference between the time the federated server submits a stored procedure to the data source, and the time it takes the data source to respond, indicating that the stored procedure has been processed.

Usage Use this element to determine how much actual time is spent at this data source processing stored procedures.

stored_procs - Stored Procedures

This element contains a count of the total number of stored procedures from the start of the federated server instance, or the last reset of the database monitor counters, that the federated server has called at this data source on behalf of any application.

Table 1258. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

Usage Use this element to determine how many stored procedure calls were made locally at the federated database or by an application against the federated database.

swap_pages_in - Pages swapped in from disk monitor element

The number of pages swapped in from disk since system startup. Reported for AIX and Linux systems only.

Table 1259. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

swap_pages_out - Pages swapped out to disk monitor element

The number of pages swapped out to disk since since system startup. Reported for AIX and Linux systems only.

Table 1260. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

swap_page_size - Swap page size monitor element

The page size used for swap space, in bytes. Reported for AIX and Linux systems only.

Table 1261. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

sync_runstats – Total number of synchronous RUNSTATS activities monitor element

The total number of synchronous RUNSTATS activities triggered by real-time statistics gathering for all the applications in the database. This value includes both successful and unsuccessful synchronous RUNSTATS commands. Values reported by all the database partitions are aggregated together.

Table 1262. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Statement

For snapshot monitoring, this counter can be reset.

Table 1263. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage

Use this monitor element to determine how many synchronous RUNSTATS activities have been triggered by real-time statistics gathering in the database. This value changes frequently. In order to get a better view of the system usage, take a snapshot at specific intervals over an extended period of time. When used in conjunction with **sync_runstats_time**, this element can help you evaluate the performance impact of synchronous RUNSTATS activities triggered by real-time statistics gathering.

sync_runstats_time – Total time spent on synchronous RUNSTATS activities monitor element

The total time spent on synchronous RUNSTATS activities triggered by real-time statistics gathering, in milliseconds. The synchronous RUNSTATS activities occur during query compilation. At the database level, this monitor element represents the total time spent on synchronous RUNSTATS activities for all the applications running on the database, triggered by real-time statistics gathering. At the statement level, it represents the time spent on the latest synchronous RUNSTATS activities for a particular statement, triggered by real-time statistics gathering. Values reported by all the database partitions are aggregated together.

Table 1264. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Statement
Dynamic SQL	dynsql	Statement

For snapshot monitoring, this element can be reset.

Table 1265. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Statement	event_stmt	-

Usage

Use this element along with **sync_runstats** to evaluate the performance impact of synchronous RUNSTATS activities triggered by real-time statistics gathering, at the database level,

For dynamic SQL snapshot monitor, use this element along with **total_exec_time** and **num_executions** to evaluate the impact of synchronous RUNSTATS on query performance.

For the statement event monitor, use this element along with **stmt_start** and **stmt_stop** for further evaluation of the impact of real-time statistics gathering.

system_auth_id - System authorization identifier monitor element

The system authorization id for the connection.

Table 1266. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	Always collected
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected

Table 1267. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Threshold violations	event_thresholdviolations	-

system_cpu_time - System CPU time monitor element

The total *system* CPU time (in seconds and microseconds) used by the database manager agent process, the unit of work, or the statement. For event monitors that write to tables, the value of this element is given in microseconds by using the BIGINT data type.

When either the statement monitor switch or the timestamp switch is not turned on, this element is not collected. In that case, the monitor element displays -1 instead.

Table 1268. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-
Transactions	event_xact	-
Statements	event_stmt	-
Activities	event_activity	-

Usage

This element, along with the other related CPU-time elements, can help you understand the level of activity within an application, and can help you identify applications that could benefit from additional tuning.

Note: If this information is not available for your operating system, this element will be set to 0.

Note: Due to the differences in granularity with which the DB2 system collects statistics, the value of the **total_exec_time** monitor element might not equal the sum of values of **system_cpu_time** and **user_cpu_time** monitor elements. In this case, the sum of **system_cpu_time** and **user_cpu_time** monitor elements more accurately reflects the actual total execution time.

tab_file_id - Table file ID monitor element

A file ID (FID) for the table.

Table 1269. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLE_METRICS table function - Get table metrics	Always collected

Usage

tab_type - Table type monitor element

This interface returns a text identifier based on defines in sqlmon.h, and is one of USER_TABLE, TEMP_TABLE, or CATALOG_TABLE.

Table 1270. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLE_METRICS table function - Get table metrics	Always collected

Usage

table_file_id - Table file ID monitor element

The file ID (FID) for the table.

Table 1271. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLE table function - Get table metrics	Always collected
MON_GET_LOCKS table function - list all locks in the currently connected database	Always collected
MON_GET_APPL_LOCKWAIT table function - get information about locks for which an application is waiting	Always collected

Table 1272. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Lock
Table	table	Basic
Lock	appl_lock_list	Lock
Lock	lock	Lock

Table 1273. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	lock	-

Usage

For snapshot monitoring, this element is provided for information purposes only. It is returned for compatibility with previous versions of the database system monitor, and it may not uniquely identify the table. Use **table_name** and **table_schema** monitor elements to identify the table.

In MON_GET_LOCKS and MON_GET_APPL_LOCKWAIT table functions, this element represents the file ID (FID) for the table that the lock references.

table_name - Table name monitor element

The name of the table.

Table 1274. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLE table function - Get table metrics	Always collected
MON_GET_INDEX table function - Get index metrics	Always collected

Table 1275. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table	Basic
Application	appl	Lock
Lock	appl_lock_list	Lock
Lock	lock	Lock
Lock	lock_wait	Lock

Table 1276. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Tables	event_table	-
Deadlocks[1]	lock	-
Deadlocks[1]	event_dlconn	-
Deadlocks with Details[1]	event_detailed_dlconn	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

Along with **table_schema**, this element can help you determine the source of contention for resources.

At the application-level, application-lock level, and deadlock-monitoring-level, this is the table that the application is waiting to lock, because it is currently locked by another application. For snapshot monitoring, this item is only valid when the "lock" monitor group information is set to ON, and when **lock_object_type** indicates that the application is waiting to obtain a table lock.

For snapshot monitoring at the object-lock level, this item is returned for table-level and row-level locks. The table reported at this level is the table against which this application holds these locks.

For snapshot and event monitoring at the table level, this is the table for which information has been collected. For temporary tables, the format for **table_name** is "TEMP (*n*, *m*)", where:

- *n* is the table space ID
- *m* is the **table_file_id** element

table_scans - Table scans monitor element

The number of scans on this table.

Table 1277. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLE table function - Get table metrics	Always collected

Usage

table_schema - Table schema name monitor element

The schema of the table.

Table 1278. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLE table function - Get table metrics	Always collected
MON_GET_INDEX table function - Get index metrics	Always collected

Table 1279. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table	Basic
Application	appl	Lock
Lock	appl_lock_list	Lock
Lock	lock	Lock
Lock	lock_wait	Lock

Table 1280. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Tables	event_table	-

Table 1280. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks[1]	lock	-
Deadlocks[1]	event_dlconn	-
Deadlocks with Details[1]	event_detailed_dlconn	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

Along with **table_name**, this element can help you determine the source of contention for resources.

For application-level, application-lock-level, deadlock-monitoring-level, this is the schema of the table that the application is waiting to lock, because it is currently locked by another application. This element is only set if **lock_object_type** indicates that the application is waiting to obtain a table lock. For snapshot monitoring at the application-level and application-lock levels, this item is only valid when the "lock" monitor group information is set to ON.

For snapshot monitoring at the object-lock level, this item is returned for table and row level locks. The table reported at this level is the table against which this application holds these locks.

For snapshot and event monitoring at the table level, this element identifies the schema of the table for which information has been collected. For temporary tables, the format for **table_schema** is "<*agent_id*><*auth_id*>", where:

- *agent_id* is the Application Handle of the application creating the temporary table
- *auth_id* is the authorization ID used by the application to connect to the database

table_type - Table type monitor element

The type of table for which information is returned.

Table 1281. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLE table function - Get table metrics	Always collected

Table 1282. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table	Basic

Table 1283. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Tables	event_table	-

Usage

Use this element to help identify the table for which information is returned. If the table is a user table or a system catalog table, you can use **table_name** and **table_schema** to identify the table.

The type of table may be one of the following. The possible values are text strings based on defines in the `sqlmon.h` file.

USER_TABLE
User table.

TEMP_TABLE
Temporary table. Information regarding temporary tables is returned, even though the tables are not kept in the database after being used. You may still find information about this type of table useful.

CATALOG_TABLE
System catalog table.

tablespace_auto_resize_enabled - Table space automatic resizing enabled monitor element

This element describes whether automatic resizing is enabled for the table space. A value of 1 means "Yes"; a value of 0 means "No".

Table 1284. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1285. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Basic

Usage

This element is only applicable to DMS table spaces and non-temporary automatic storage table spaces. If this element is set to 1, then automatic resizing is enabled. See the following monitor elements for information about the rate of increase and the maximum size for the table space.

- **tablespace_max_size**
- **tablespace_increase_size**
- **tablespace_increase_size_percent**

tablespace_content_type - Table space content type monitor element

The type of content in a table space.

Table 1286. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1287. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Basic

Usage

The type of content in the table space (defined in sqlmon.h) can be one of the following:

- All types of permanent data.
  - Regular table space: SQLM_TABLESPACE_CONTENT_ANY
  - Large table space: SQLM_TABLESPACE_CONTENT_LARGE
- System temporary data: SQLM_TABLESPACE_CONTENT_SYSTEMP
- User temporary data: SQLM_TABLESPACE_CONTENT_USRTEMP

tablespace_cur_pool_id - Buffer pool currently being used monitor element

The buffer pool identifier for a buffer pool that a table space is currently using.

Table 1288. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1289. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Basic

Usage Each buffer pool is identified by a unique integer. The value of this element matches a value from column BUFFERPOOLID of view SYSCAT.BUFFERPOOLS.

tablespace_current_size - Current table space size

This element shows the current size of the table space in bytes.

Table 1290. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage For DMS and automatic storage table spaces, this element represents the total size of all table space containers in bytes. This value is equal to the total pages for the table space (tablespace_total_pages) multiplied by the table space's page size (tablespace_page_size). This element is not applicable for SMS table spaces, or for temporary automatic storage table spaces.

On table space creation for an automatic storage table space, the current size might not match the initial size. The value of current size will be within page size multiplied by extent size multiplied by the number of storage paths of the initial size on creation (usually greater, but sometimes smaller). It will always be less than or equal to tablespace_max_size (if set). This is because containers can only grow by full extents, and must be grown as a set.

tablespace_extent_size - Table space extent size monitor element

The extent size used by a table space.

Table 1291. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1292. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Basic

tablespace_free_pages - Free pages in table space monitor element

The total number of pages that are currently free in a table space.

Table 1293. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1294. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage

This is applicable only to a DMS table space.

tablespace_id - Table space identification monitor element

An integer that uniquely represents a table space used by the current database.

Table 1295. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_CONTAINER table function - Get table space container metrics	Always collected
MON_GET_TABLE table function - Get table metrics	Always collected
MON_GET_TABLESPACE table function - Get table space metrics	Always collected
MON_GET_EXTENT_MOVEMENT_STATUS - Get extent movement progress status metrics	Always collected
MON_GET_LOCKS table function - list all locks in the currently connected database	Always collected
MON_GET_APPL_LOCKWAIT table function - get information about locks for which an application is waiting	Always collected

Table 1296. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Basic
Table	table	Basic

Table 1297. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Tables	event_table	-

Usage

The value of this element matches a value from column TBSPACEID of view SYSCAT.TABLESPACES.

tablespace_increase_size - Increase size in bytes

This element shows the size that an auto-resize table space will increase by in bytes when the table space becomes full and more space is required.

Table 1298. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage This represents the amount of space that will be added to a table space that can be automatically resized when it becomes full, more space is being requested, and the maximum table space size has not been reached. If the value of this element is -1 (or "AUTOMATIC" in the snapshot output),

then DB2 automatically determines the value when space needs to be added. This element is only applicable to table spaces that are enabled to be automatically resized.

tablespace_increase_size_percent - Increase size by percent monitor element

This element shows the amount by which an auto-resize table space will increase when the table space becomes full and more space is required. The actual number of bytes is determined at the time the table space is resized based on the size of the table space at that time.

Table 1299. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage This represents the amount of space that will be added to a table space that can be automatically resized when it becomes full, more space is being requested, and the maximum table space size has not been reached. The growth rate is based on a percentage of the current table space size (tablespace_current_size) at the time the table space is resized. This element is only applicable to table spaces that are enabled to be automatically resized.

tablespace_initial_size - Initial table space size

The initial size of the automatic storage table space in bytes.

Table 1300. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage For non-temporary automatic storage table spaces, this monitor element represents the initial size in bytes for the table space when it was created.

tablespace_last_resize_failed - Last resize attempt failed

This element describes whether or not the last attempt to automatically increase the size of the table space failed. A value of 1 means yes, 0 means no.

Table 1301. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage For an automatic storage table space, this element may show that there is no space left on any of the database's storage paths. For a non-automatic storage table space, a failure means that one of the containers could not be extended because its filesystem was full. Another reason for failure is that the maximum size of the table space has been reached. This element is only applicable to table spaces that are enabled to be automatically resized.

tablespace_last_resize_time - Time of last successful resize

This element shows a timestamp representing the last time that the size of the table space was successfully increased.

Table 1302. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage For table spaces that can be automatically resized, this element represents the last time that space was automatically added to the table space when it became full, more space was being requested, and the maximum table space size had not been reached. This element is only applicable to table spaces that are enabled to be automatically resized.

tablespace_max_size - Maximum table space size

This element shows the maximum size in bytes to which the table space can automatically resize or increase.

Table 1303. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage This represents the maximum size in bytes to which a table space that can be automatically resized can automatically increase. If this value is equal to the tablespace_current_size element, then there is no room for the table space to grow. If the value of this element is -1, then the maximum size is considered to be "unlimited" and the table space can automatically resize until the file systems are full or the architectural size limit of the table space is reached. (This limit is described in the SQL Limits appendix of the *SQL Reference*). This element is only applicable to table spaces that are enabled for automatic resizing.

tablespace_min_recovery_time - Minimum recovery time for rollforward monitor element

A timestamp showing the earliest point in time to which a table space can be rolled forward. The timestamp reflects local time.

Table 1304. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage

Displayed only if non zero.

tablespace_name - Table space name monitor element

The name of a table space.

Table 1305. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONTAINER table function - Get table space container metrics	Always collected
MON_GET_TABLESPACE table function - Get table space metrics	Always collected
MON_GET_EXTENT_MOVEMENT_STATUS - Get extent movement progress status metrics	Always collected

Table 1306. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Basic
Lock	appl_lock_list	Basic
Lock	lock	Lock
Lock	lock_wait	Lock

Table 1307. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks[1]	lock	-
Deadlocks[1]	event_dlconn	-
Deadlocks with Details[1]	event_detailed_dlconn	-
Table Space	tablespace_list	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

Usage

This element can help you determine the source of contention for resources.

It is equivalent to the TBSPACE column in the database catalog table SYSCAT.TABLESPACES. At the application level, application-lock level, and deadlock monitoring level, this is the name of the table space that the application is waiting to lock. Another application currently holds a lock on this table space.

At the lock level, this is the name of the table space against which the application currently holds a lock.

At the table space level (when the buffer pool monitor group is ON), this is the name of the table space for which information is returned.

This element will not be returned for a table lock held on a partitioned table.

tablespace_next_pool_id - Buffer pool that will be used at next startup monitor element

The buffer pool identifier for a buffer pool that a table space will use at the next database startup.

Table 1308. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1309. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Basic

Usage Each buffer pool is identified by a unique integer. The value of this element matches a value from column BUFFERPOOLID of view SYSCAT.BUFFERPOOLS

tablespace_num_containers - Number of Containers in Table Space

Total number of containers in the table space.

Table 1310. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

tablespace_num_quiescers - Number of Quiescers

The number of users quiescing the table space (can be in the range of 0 to 5).

Table 1311. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage This value represents the number of agents that have quiesced the table space (either in "SHARE", "UPDATE", or "EXCLUSIVE" mode). For each quiescer, the following information is returned in a tablespace_quiescer logical data group:

- User authorization ID of the quiescer
- Agent ID of the quiescer
- Table space ID of the object that was quiesced that resulted in this table space being quiesced
- Object ID of the object that was quiesced that resulted in this table space being quiesced
- Quiesce state

tablespace_num_ranges - Number of Ranges in the Table Space Map

The number of ranges (entries) in the table space map. This can be in the range of 1 to 100's (but is usually less than a dozen). The table space map only exists for DMS table spaces.

Table 1312. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

tablespace_page_size - Table space page size monitor element

Page size used by a table space in bytes.

Table 1313. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1314. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Basic

tablespace_page_top - Table space high watermark monitor element

The page in a table space that is holding the high watermark.

Table 1315. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1316. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage

For DMS, this element represents the page number of the first free extent following the last allocated extent of a table space. Note that this is not really a "high watermark", but rather a "current watermark", since the value can decrease. For SMS, this is not applicable.

tablespace_paths_dropped - Table space using dropped path monitor element

Indicates that the table space is using a storage path that has been dropped.

Table 1317. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1318. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage

For table spaces using automatic storage, use this monitor element to determine whether any of the table space containers reside on a storage path that has been dropped. Before storage paths are physically dropped from the database, all table spaces must stop using them. To stop using a dropped storage path, either drop the table space or rebalance the table space using the REBALANCE clause of the ALTER TABLESPACE statement.

tablespace_pending_free_pages - Pending free pages in table space monitor element

The number of pages in a table space which would become free if all pending transactions are committed or rolled back and new space is requested for an object.

Table 1319. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1320. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage

This is applicable only to a DMS table space.

tablespace_prefetch_size - Table space prefetch size monitor element

The maximum number of pages the prefetcher gets from the disk at a time.

Table 1321. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1322. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Basic
Table Space	tablespace_nodeinfo	Basic

Usage

- For table function monitoring, this element always reports the actual value for the table space prefetch size.
- For snapshot monitoring, if automatic prefetch size is enabled, this element reports the value "-1" in the *tablespace* Logical Data Grouping, and the actual value is reported in the *tablespace_nodeinfo* Logical Data Grouping.
- For snapshot monitoring, if automatic prefetch size is not enabled, this element reports the actual value in the *tablespace* Logical Data Grouping, and the element does not appear in the *tablespace_nodeinfo* Logical Data Grouping.

tablespace_rebalancer_extents_processed - Number of Extents the Rebalancer has Processed

The number of extents that the rebalancer has already moved since the rebalancer has been started or restarted (whichever is most recent).

Element identifier
tablespace_rebalancer_extents_processed

Element type
information

Table 1323. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage This can be used as an indicator of the completion level of the rebalancer. You can monitor the progress of rebalancing by keeping track of the change in this element over time. You can use tablespace_state and rebalance_mode to check if the rebalancing is completed. This is only applicable to a DMS table space.

tablespace_rebalancer_extents_remaining - Total Number of Extents to be Processed by the Rebalancer

The number of extents to be moved. This value is calculated at either the rebalancer start time or restart time (whichever is most recent).

Element identifier
tablespace_rebalancer_extents_remaining

Element type
information

Table 1324. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage This can be used as an indicator of the completion level of the rebalancer. You can monitor the progress of rebalancing by keeping track of the change in this element over time. You can use tablespace_state to check if rebalancing has completed. This is only applicable to a DMS table space.

tablespace_rebalancer_last_extent_moved - Last Extent Moved by the Rebalancer

The last extent moved by the rebalancer.

Element identifier
tablespace_rebalancer_last_extent_moved

Element type
information

Table 1325. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage This can be used as an indicator of the completion level of the rebalancer. You can monitor the progress of rebalancing by keeping track of the change in this element over time. You can use tablespace_state and rebalance_mode to check if the rebalancing is completed. This is only applicable to a DMS table space.

tablespace_rebalancer_mode - Rebalancer mode monitor element

Indicates whether the current rebalance process is removing space from a table space or adding space to a table space.

Table 1326. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	Always collected

Table 1327. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage

A *forward rebalance* occurs when new containers are added or existing containers have their size increased. In a forward rebalancing operation, data movement starts with the first extent in the table space and ends with the high watermark extent.

A *reverse rebalance* occurs when containers are removed or reduced in size and data needs to move out of the space being freed. In a reverse rebalancing operation, data movement starts at the high watermark extent and moves in reverse order through the table space, ending with the first extent in the table space.

A *two-pass rebalance* is a forward rebalance followed by a reverse rebalance. A two-pass rebalance might occur when containers are being both added and dropped as part of the rebalance operation.

For DMS non-automatic storage table spaces, this monitor element indicates the type of rebalance that is occurring for the table space. Only a single forward rebalance or a single reverse rebalance can occur for DMS non-automatic table space.

For automatic storage table spaces, this monitor element indicates what the current rebalance process is doing to the table space. In general, only a single forward rebalance or a single reverse rebalance is necessary when a rebalance is initiated. However, there are cases when a two-pass rebalance is necessary for automatic storage table spaces.

The possible **tablespace_rebalancer_mode** values are defined in the `sqlmon.h` file. The following values are returned in snapshot monitoring:

SQLM_TABLESPACE_NO_REBAL
No rebalancing is taking place.

SQLM_TABLESPACE_FWD_REBAL
Forward rebalance is taking place.

SQLM_TABLESPACE_REV_REBAL
Reverse rebalance is taking place.

SQLM_TABLESPACE_FWD_REBAL_OF_2PASS
The forward rebalance phase of two pass rebalance is taking place.

SQLM_TABLESPACE_REV_REBAL_OF_2PASS
The reverse rebalance phase of two pass rebalance is taking place.

If using the MON_GET_TABLESPACE table function, the following rebalancer_mode values are returned:

- NO_REBAL
- FWD_REBAL
- REV_REBAL
- FWD_REBAL_OF_2PASS
- REV_REBAL_OF_2PASS

tablespace_rebalancer_priority - Current Rebalancer Priority

The priority at which the rebalancer is running in the database.

Table 1328. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage This is only applicable to a DMS table space.

tablespace_rebalancer_restart_time - Rebalancer Restart Time

A timestamp representing when a rebalancer was restarted after being paused or stopped.

Element identifier
tablespace_rebalancer_restart_time

Element type
information

Table 1329. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage This can be used as an indicator of the completion level of the rebalancer. It will note when the rebalancer was restarted, and will allow for the derivation of the speed of the rebalancer and the estimated time until completion. This is only applicable to a DMS table space.

tablespace_rebalancer_start_time - Rebalancer Start Time

A timestamp representing when a rebalancer was initially started.

Element identifier
tablespace_rebalancer_start_time

Element type
information

Table 1330. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage This will be used to note the time at which a rebalancer was initially started. This can be used to derive metrics as to the speed at which the rebalancer is operating, and the estimated time of completion of the rebalance. This is only applicable to a DMS table space.

tablespace_state - Table space state monitor element

This element describes the current state of a table space.

Table 1331. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1332. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage

In administrative views and table functions, this monitor element returns a text identifier based on defines in `sqlutil.h`, and is combination of the following separated by a '+' sign:

- BACKUP_IN_PROGRESS
- BACKUP_PENDING
- DELETE_PENDING
- DISABLE_PENDING
- DROP_PENDING
- LOAD_IN_PROGRESS
- LOAD_PENDING
- MOVE_IN_PROGRESS
- NORMAL
- OFFLINE
- PSTAT_CREATION
- PSTAT_DELETION
- QUIESCED_EXCLUSIVE
- QUIESCED_SHARE
- QUIESCED_UPDATE
- REBAL_IN_PROGRESS
- REDIST_IN_PROGRESS
- REORG_IN_PROGRESS
- RESTORE_IN_PROGRESS
- RESTORE_PENDING
- ROLLFORWARD_IN_PROGRESS
- ROLLFORWARD_PENDING
- STORDEF_ALLOWED
- STORDEF_CHANGED
- STORDEF_FINAL_VERSION
- STORDEF_PENDING
- SUSPEND_WRITE

This element contains a hexadecimal value indicating the current table space state. The externally visible state of a table space is composed of the hexadecimal sum of certain state values. For example, if the state is "quiesced: EXCLUSIVE" and "Load pending", the value is 0x0004 + 0x0008, which is 0x000c. Use the **db2tbst** command to obtain the table space state associated with a given hexadecimal value.

Table 1333. Bit definitions listed in sqlutil.h

Hexadecimal Value	Decimal Value	State
0x0	0	Normal (see the definition SQLB_NORMAL in sqlutil.h)
0x1	1	Quiesced: SHARE
0x2	2	Quiesced: UPDATE
0x4	4	Quiesced: EXCLUSIVE
0x8	8	Load pending
0x10	16	Delete pending
0x20	32	Backup pending
0x40	64	Roll forward in progress
0x80	128	Roll forward pending
0x100	256	Restore pending
0x100	256	Recovery pending (not used)
0x200	512	Disable pending
0x400	1024	Reorg in progress
0x800	2048	Backup in progress
0x1000	4096	Storage must be defined
0x2000	8192	Restore in progress
0x4000	16384	Offline and not accessible
0x8000	32768	Drop pending
0x10000	65536	No write is allowed
0x20000	131072	Load in progress
0x40000	262144	Redistribute in progress
0x80000	524288	Move in progress
0x2000000	33554432	Storage may be defined
0x4000000	67108864	Storage Definition is in 'final' state
0x8000000	134217728	Storage Definition was changed prior to rollforward
0x10000000	268435456	DMS rebalancer is active
0x20000000	536870912	TBS deletion in progress
0x40000000	1073741824	TBS creation in progress

Note: DB2 LOAD does not set the table space state to `Load pending` or `Delete pending`.

tablespace_state_change_object_id - State Change Object Identification

The object that caused the table space state to be set to "Load pending" or "Delete pending".

Element identifier
tablespace_state_change_object_id

Element type
information

Table 1334. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage This element is meaningful only if the table space state is "Load pending" or "Delete pending". If nonzero, the value of this element matches a value from column TABLEID of view SYSCAT.TABLES.

Note: DB2 LOAD does not set the table space state to `Load pending` or `Delete pending`.

tablespace_state_change_ts_id - State Change Table Space Identification

If the table space state is "Load pending" or "Delete pending", this shows the table space ID of the object that caused the table space state to be set.

Element identifier
tablespace_state_change_ts_id

Element type
information

Table 1335. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

Usage This element is meaningful only if the table space state is "Load pending" or "Delete pending". If nonzero, the value of this element matches a value from column TABLESPACEID of view SYSCAT.TABLES.

Note: DB2 LOAD does not set the table space state to `Load pending` or `Delete pending`.

tablespace_total_pages - Total pages in table space monitor element

Total number of pages in a table space.

Table 1336. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1337. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic (DMS table spaces) Buffer Pool (SMS table spaces)

Usage

Total operating system space occupied by a table space. For DMS, this is the sum of the container sizes (including overhead). For SMS, this is the sum of all file space used for the tables stored in this table space (and is only collected if the buffer pool switch is on).

tbsp_trackmod_state - Table space trackmod state monitor element

The modification state that a table space is in with respect to the last or next backup.

Table 1338. Table function monitoring information

Table function	Monitor element collection level
MON_GET_TABLESPACE table function - Get table space metrics	Always

Usage

You can use this monitor element to determine the modification status of a table space. The status of a table space can be in one of the following states:

CLEAN
No modifications occurred in the tablespace since the previous backup. If an incremental or delta backup is executed at this time, no data pages from this tablespace would be backed up.

DIRTY
Table space contains data that needs to be picked up by the next backup.

ININCREMENTAL
Table space contains modifications that were copied into an incremental backup. This state is in a `DIRTY` state relative to a full backup such that a future incremental backup needs to include some pages from this pool. This state is also in a `CLEAN` state such that a future delta backup does not need to include any pages from this pool.

READFULL
The latest table space modification state change was caused by a dirty table space that is being read by a full backup that might not have completed successfully, or is currently in progress.

READINCREMENTAL
The latest table space modification state change was caused by a dirty table space that is being read by an incremental backup that might not have completed successfully, or is currently in progress.

UNAVAILABLE
The **trackmod** configuration parameter is set to `No`. Therefore, no table space modification status information is available.

tablespace_type - Table space type monitor element

The type of a table space.

Table 1339. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1340. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Basic

Usage

This element shows whether this table space is a database managed table space (DMS), or system managed table space (SMS).

The values for tablespace_type (defined in sqlmon.h) are as follows:

- For DMS: SQLM_TABLESPACE_TYP_DMS
- For SMS: SQLM_TABLESPACE_TYP_SMS

tablespace_usable_pages - Usable pages in table space monitor element

The total number of pages in a table space minus overhead pages.

Table 1341. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1342. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic (DMS table spaces) Buffer Pool (SMS table spaces)

Usage

This element is applicable to DMS table spaces only. For SMS table spaces, this element will have the same value as the **tablespace_total_pages** monitor element.

During a table space rebalance, the number of usable pages will include pages for the newly added container, but these new pages may not be reflected in the number of free pages until the rebalance is complete. When a table space rebalance is not taking place, the number of used pages plus the number of free pages, plus the number of pending free pages will equal the number of usable pages.

tablespace_used_pages - Used pages in table space monitor element

The total number of pages that are currently used (not free) in a table space.

Table 1343. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1344. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic (DMS table spaces) Buffer Pool (SMS table spaces)

Usage

This is the total number of pages in use for a DMS table space. For an SMS table space it is equal to the value of **tablespace_total_pages** monitor element.

tablespace_using_auto_storage - Table space enabled for automatic storage monitor element

This element describes whether the table space was created as an automatic storage table space. A value of 1 means "Yes"; a value of 0 means "No".

Table 1345. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1346. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Basic

Usage

You can use this element to determine whether the given table space was created using automatic storage (that is, created with the MANAGED BY AUTOMATIC STORAGE clause), rather than with containers that are explicitly provided. The table space can have containers that exist on some or all of the storage paths associated with the database.

tbsp_max_page_top - Maximum table space page high watermark monitor element

The highest allocated page number for a DMS table space since the database was activated.

Table 1347. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Usage

This value changes whenever the value of the **tablespace_page_top** monitor element increases.

tcpip_recv_volume - TCP/IP received volume monitor element

The amount of data received by the data server from clients over TCP/IP. This value is reported in bytes.

Table 1348. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1349. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE

Table 1349. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

tcpip_recv_wait_time - TCP/IP received wait time monitor element

The time spent waiting for an incoming client request over TCP/IP excluding idle time. The value is given in milliseconds.

Table 1350. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1351. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

Table 1351. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

tcpip_recvs_total - TCP/IP receives total monitor element

The number of times data was received by the database server from the client application over TCP/IP.

Table 1352. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1353. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

Table 1353. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.		

tcpip_send_volume - TCP/IP send volume monitor element

The amount of data sent by data server to client. This value is reported in bytes.

Table 1354. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1355. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

tcpip_send_wait_time - TCP/IP send wait time monitor element

Time spent blocking on a TCP/IP send to the client. The value is given in milliseconds.

Table 1356. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1357. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

tcpip_sends_total - TCP/IP sends total monitor element

The number of times data was sent from the database server to the client application over TCP/IP.

Table 1358. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1359. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

temp_tablespace_top - Temporary table space top monitor element

The high watermark in KB for the temporary table space usage of DML activities at all nesting levels in a service class or work class. For service classes, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service class is set to NONE. For work classes, this monitor elements returns -1 if no COLLECT AGGREGATE ACTIVITY DATA work action is specified for the

work class. For workloads, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the workload is set to NONE.

For service classes, when you remap activities between service subclasses with a REMAP ACTIVITY action, only the temp_tablespace_top high watermark of the service subclass where an activity completes is changed. High watermarks of service subclasses an activity is mapped to but does not complete in are unaffected.

Table 1360. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wcstats	-
Statistics	event_wlstats	-

Usage

Use this element to determine the highest DML activity system temporary table space usage reached on a partition for a service class, workload, or work class in the time interval collected.

This element is only updated by activities that have a temporary table space threshold applied to them. If no temporary table space threshold is applied to an activity, a value of 0 is returned.

territory_code - Database Territory Code

The territory code of the database for which the monitor data is collected. This monitor element was formerly known as country_code.

Table 1361. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
Application	appl	Basic

Table 1362. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Event Log Header	event_log_header	-
Connections	event_connheader	-

Usage Territory code information is recorded in the database configuration file.

For DRDA AS connections, this element will be set to 0.

thresh_violations - Number of threshold violations monitor element

Number of times a threshold was violated.

This monitor element is an alias of the “num_threshold_violations - Number of threshold violations monitor element” on page 780 monitor element, which is returned by snapshot monitoring routines and the Database event monitor.

Table 1363. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1364. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element to quickly determine if there have been any WLM thresholds that have been violated. If thresholds have been violated you can then use the threshold violations event monitor (if created and active) to obtain details about the threshold violations.

For example, to obtain details which threshold was violated.

threshold_action - Threshold action monitor element

The action of the threshold to which this threshold violation record applies. Possible values include Stop, Continue and Remap.

Table 1365. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Threshold violations	event_thresholdviolations	-

Usage

Use this element to determine whether the activity that violated the threshold was stopped when the violation occurred, was allowed to continue executing, or was remapped to another service subclass. If the activity was stopped, the application that submitted the activity will have received an SQL4712N error. If the activity was remapped to another service subclass, agents working for the activity on the partition will be moving to the target service subclass of the threshold.

threshold_domain - Threshold domain monitor element

The domain of the threshold responsible for this queue.

Possible values are

- Database
- Work Action Set
- Service Superclass
- Service Subclass
- Workload

Table 1366. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_qstats	-

Usage

This element can be used for distinguishing the queue statistics of thresholds that have the same predicate but different domains.

threshold_maxvalue - Threshold maximum value monitor element

For non-queuing thresholds, this monitor element represents the value that was exceeded to cause this threshold violation. For queuing thresholds, this monitor element represents the level of concurrency that caused the queuing. The level of concurrency that caused the violation of the queuing threshold is the sum of **threshold_maxvalue** and **threshold_queuesize** monitor elements.

Table 1367. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Threshold violations	event_thresholdviolations	-

Usage

For activity thresholds, this element provides a historical record of what the threshold's maximum value was at the time the threshold was violated. This is useful when the threshold's maximum value has changed since the time of the violation and the old value is no longer available from the SYSCAT.THRESHOLDS view.

threshold_name - Threshold name monitor element

The unique name of the threshold responsible for this queue.

Table 1368. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_qstats	-

Usage

Use this element to uniquely identify the queuing threshold whose statistics this record represents.

threshold_predicate - Threshold predicate monitor element

Identifies the type of threshold that was violated or for which statistics were collected.

Table 1369. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Threshold violations	event_thresholdviolations	-
Statistics	event_qstats	-

Usage

Use this monitor element in conjunction with other statistics or threshold violation monitor elements for analysis of a threshold violation.

threshold_queuesize - Threshold queue size monitor element

The size of the queue for a queuing threshold. An attempt to exceed this size causes a threshold violation. For a non-queuing threshold, this value is 0.

Table 1370. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Threshold violations	event_thresholdviolations	-

Usage

Use this element to determine the number of activities or connections in the queue for this threshold at the time the threshold was violated.

thresholdid - Threshold ID monitor element

Identifies the threshold to which a threshold violation record applies or for which queue statistics were collected.

Table 1371. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Threshold violations	event_thresholdviolations	-
Statistics	event_qstats	-

Usage

Use this monitor element in conjunction with other activity history monitor elements for analysis of a threshold queue or for analysis of the activity that violated a threshold.

time_completed - Time completed monitor element

The time at which the activity described by this activity record finished executing. This element is a local timestamp.

Table 1372. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

Usage

Use this element in conjunction with other activity history elements for analysis of the behavior of an activity.

This field has a value of "0000-00-00-00.00.00.000000" when a full activity record could not be written to a table event monitor due to memory limitations. If the activity was captured while it was in progress, then this field represents the time that activity was collected.

time_created - Time created monitor element

The time at which a user submitted the activity described by this activity record. This element is a local timestamp.

Table 1373. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

Usage

Use this element in conjunction with other activity history elements for analysis of the behavior of an activity.

time_of_violation - Time of violation monitor element

The time at which the threshold violation described in this threshold violation record occurred. This element is a local timestamp.

Table 1374. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Threshold violations	event_thresholdviolations	-

Usage

Use this element in conjunction with other threshold violations monitor elements for analysis of a threshold violation.

time_stamp - Snapshot Time

The date and time when the database system monitor information was collected.

Table 1375. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	collected	Basic

Usage You can use this element to help relate data chronologically if you are saving the results in a file or database for ongoing analysis.

time_started - Time started monitor element

The time at which the activity described by this activity record began executing. This element is a local timestamp.

Table 1376. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

Usage

Use this element in conjunction with other activity history elements for analysis of the behavior of an activity.

If the activity got rejected, then the value of **act_exec_time** monitor element is 0. In this case, the value of **time_started** monitor element equals the value of **time_completed** monitor element.

time_zone_disp - Time Zone Displacement

Number of seconds that the local time zone is displaced from Greenwich Mean Time (GMT).

Table 1377. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	collected	Basic

Usage All time reported by reported by the database system monitor is GMT, this displacement calculates the local time.

top - Histogram bin top monitor element

The inclusive top end of the range of a histogram bin. The value of this monitor element is also the bottom exclusive end of the range of the next histogram bin.

Table 1378. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_histogrambin	-

Usage

Use this element with the corresponding **bottom** element to determine the range of a bin within a histogram.

tot_log_used_top - Maximum Total Log Space Used

The maximum amount of total log space used (in bytes).

Table 1379. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Table 1380. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage You can use this element to help evaluate the amount of primary log space that you have allocated. Comparing the value of this element with the amount of primary log space you have allocated can help you to evaluate your configuration parameter settings. Your primary log space allocation can be calculated using the following formula:

```
logprimary x logfilsiz x 4096 (see note below)
```

You can use this element in conjunction with *sec_log_used_top* and *sec_logs_allocated* to show your current dependency on secondary logs.

This value includes space used in both primary and secondary log files.

You may need to adjust the following configuration parameters:

- logfilsiz
- logprimary
- logsecond

Note: While the database system monitor information is given in bytes, the configuration parameters are set in pages, which are each 4K bytes.

total_act_time - Total activity time monitor element

The total amount of time spent executing activities. This value is given in milliseconds.

Table 1381. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1382. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE

Usage

Use this monitor element along with the **total_act_wait_time** monitor element to determine the percentage of time the data server spent working on the activity.

```
(total_act_time - total_act_wait_time)/(total_act_time) =
 % of time data server is actively working on activity
```

total_act_wait_time - Total activity wait time monitor element

Total time spent waiting within the DB2 database server, while processing an activity. The value is given in milliseconds.

Table 1383. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1383. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1384. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE

Usage

Use this monitor element along with the **total_act_time** monitor element to determine the percentage of time the data server spent working on the activity.

```
(total_act_time - total_act_wait_time)/(total_act_time) =
 % of time data server is actively working on activity
```

total_app_commits - Total application commits monitor elements

Total number of commit statements issued by the client application.

Table 1385. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE

Table 1385. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1386. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

total_app_rollbacks - Total application rollbacks monitor element

Total number of rollback statements issued by the client application.

Table 1387. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1388. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

total_app_rqst_time - Total application request time monitor element

The total elapsed time spent on application requests; this is the total time spent by coordinator agents on the server executing application requests. This value is reported in milliseconds.

Table 1389. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Usage

Use this monitor element to determine the time that the application request spent in the DB2 data server. This value can be used to help determine if the data server is the source of an observed performance problem.

For example, if a user reports that there is a problem with an application and it has taken 20 minutes to return, and if you determine that total application request time is 1 minute and there are currently no application requests in progress for the connection, then the performance problem might lie outside of the DB2 data server.

total_app_section_executions - Total application section executions monitor element

Number of section executions performed by an application.

Table 1390. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1391. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE

Table 1391. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

total_buffers_rcvd - Total FCM buffers received monitor element

For snapshot monitor, this monitor element reports the total number of FCM buffers received by the node issuing the GET SNAPSHOT command from the node identified by the **node_number** monitor element. For table function monitor, this monitor element reports the total number of FCM buffers received from a remote database member.

Table 1392. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_FCM_CONNECTION_LIST - Get details for all FCM connections	Always collected

Table 1393. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	fcm_node	Basic

Usage

Use this element to measure the level of traffic between the current member and the remote member. If the total number of FCM buffers received from this member is high, consider redistributing the database or moving tables to reduce the traffic between members.

total_buffers_sent - Total FCM buffers sent monitor element

For snapshot monitor, this monitor element reports the total number of FCM buffers that have been sent from the node issuing the GET SNAPSHOT command to the node identified by the **node_number** monitor element. For table function monitor, this monitor element reports the total number of FCM buffers sent from the current database member to a remote database member.

Table 1394. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_FCM_CONNECTION_LIST - Get details for all FCM connections	Always collected

Table 1395. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	fcm_node	Basic

Usage

Use this element to measure the level of traffic between the current member and the remote member. If the total number of FCM buffers sent to this member is high, consider redistributing the database or moving tables to reduce the traffic between members.

total_bytes_received - Bytes received monitor element

Total number of bytes received since the network adapter started.

Table 1396. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_NETWORK_RESOURCES table function - Return network adapter information	Always collected

total_bytes_sent - Bytes sent monitor element

Total number of bytes sent s since the network adapter started.

Table 1397. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_NETWORK_RESOURCES table function - Return network adapter information	Always collected

total_commit_proc_time - Total commits processing time monitor element

The total amount of processing (non-wait) time spent performing commit processing on the database server. The value is given in milliseconds.

Table 1398. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1398. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1399. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

total_commit_time - Total commit time monitor element

The total amount of time spent performing commit processing on the database server. The value is given in milliseconds.

Table 1400. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 1400. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1401. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

total_compilations - Total compilations monitor element

The total number of explicit compiles on the database server. Explicit compiles are compilations directly initiated by a user request such as a bind, rebind, prepare or execute immediate.

Table 1402. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 1402. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1403. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

total_compile_proc_time - Total compile processing time monitor element

The total amount of processing (non-wait) time spent performing explicit compiles on the database server. Explicit compiles are compilations directly initiated by a user request such as a bind, rebind, prepare or execute immediate. The value is given in milliseconds.

Table 1404. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 1404. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1405. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

total_compile_time - Total compile time monitor element

The total amount of time spent performing explicit compiles on the database server. Explicit compiles are compilations directly initiated by a user request such as a bind, rebind, prepare or execute immediate. The value is given in milliseconds.

Table 1406. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 1406. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1407. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

total_cons - Connects Since Database Activation

Indicates the number of connections to the database since the first connect, activate, or last reset (coordinator agents).

Table 1408. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Database	dbase_remote	Basic

For snapshot monitoring, this counter can be reset.

Table 1409. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Usage You can use this element in conjunction with the db_conn_time and the db2start_time monitor elements to calculate the frequency at which applications have connected to the database.

If the frequency of connects is low, you may want to explicitly activate the database using the ACTIVATE DATABASE command before connecting any other application, because of the extra overhead that is associated with

the first connect to a database (for example, initial buffer pool allocation). This will result in subsequent connects being processed at a higher rate.

Note: When you reset this element, its value is set to the number of applications that are currently connected, not to zero.

total_cpu_time - Total CPU time monitor element

The total amount of CPU time used while within DB2. Represents total of both user and system CPU time. This value is given in microseconds.

Table 1410. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 1411. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE

Table 1411. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

total_exec_time - Elapsed statement execution time monitor element

The total time in seconds and microseconds that was spent executing a particular statement in the SQL cache.

Table 1412. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Dynamic SQL	dynsql	Statement

For snapshot monitoring, this counter can be reset.

Usage

Use this element with **num_executions** monitor element determine the average elapsed time for the statement and identify the SQL statements that would most benefit from a tuning of their SQL. The **num_compilation** monitor element must be considered when evaluating the contents of this element.

Note: Due to the differences in granularity with which the DB2 system collects statistics, the value of the **total_exec_time** monitor element might not equal the sum of values of **system_cpu_time** and **user_cpu_time** monitor elements. In this case, the sum of **system_cpu_time** and **user_cpu_time** monitor elements more accurately reflects the actual total execution time.

total_hash_joins - Total Hash Joins

The total number of hash joins executed.

Element identifier
total_hash_joins

Element type
counter

Table 1413. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 1414. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage At the database or application level, use this value in conjunction with hash_join_overflows and hash_join_small_overflows to determine if a significant percentage of hash joins would benefit from modest increases in the sort heap size.

total_hash_loops - Total Hash Loops

The total number of times that a single partition of a hash join was larger than the available sort heap space.

Table 1415. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 1416. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage Values for this element indicate inefficient execution of hash joins. This might indicate that the sort heap size is too small or the sort heap threshold is too small. Use this value in conjunction with the other hash join variables to tune the sort heap size (*sortheap*) and sort heap threshold (*sheapthres*) configuration parameters.

total_implicit_compilations - Total implicit complications monitor element

Total number of implicit compiles on the database server. Implicit compiles are compilations that are not directly requested by the user. That is, they are not a result of a bind, rebind, prepare or execute immediate request. For example, an implicit compilation may occur when executing a statement that was bound using the VALIDATE RUN option if the statement needs to be compiled at execution time.

Table 1417. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE

Table 1417. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1418. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

total_implicit_compile_proc_time - Total implicit compile processing time monitor element

Total amount of processing (non-wait) time spent performing implicit compiles on the database server. Implicit compiles are compilations that are not directly requested by the user. That is, they are not a result of a bind, rebind, prepare or execute immediate request. For example, an implicit compilation may occur when executing a statement that was bound using the VALIDATE RUN option if the statement needs to be compiled at execution time. The value is given in milliseconds.

Table 1419. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1420. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

total_implicit_compile_time - Total implicit compile time monitor element

The total amount of time spent performing implicit compiles on the database server. Implicit compiles are compilations that are not directly requested by the user. That is, they are not a result of a bind, rebind, prepare or execute immediate request. For example, an implicit compilation may occur when executing a statement that was bound using the VALIDATE RUN option if the statement needs to be compiled at execution time. The value is given in milliseconds.

Table 1421. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1422. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

total_load_proc_time - Total load processing time monitor element

Total amount of processing (non-wait) time spent performing load processing on the database server. The value is given in milliseconds.

Table 1423. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE

Table 1423. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1424. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

total_load_time - Total load time monitor element

The total amount of time spent performing loads on the database server. The value is given in milliseconds.

Table 1425. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1425. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1426. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

total_loads - Total loads monitor element

The total number of loads performed on the database server.

Table 1427. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 1427. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1428. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml) document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE

total_log_available - Total Log Available

The amount of active log space in the database that is not being used by uncommitted transactions (in bytes).

Table 1429. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Usage

Use this element in conjunction with total_log_used to determine whether you may need to adjust the following configuration parameters to avoid running out of log space:

- logfilsiz
- logprimary
- logsecond

If total_log_available goes down to 0, SQL0964N will be returned. You may need to increase the above configuration parameters, or end the oldest transaction by COMMIT, ROLLBACK or FORCE APPLICATION.

If logsecond is set to -1 this element will contain SQLM_LOGSPACE_INFINITE.

Note: While the database system monitor information is given in bytes, the configuration parameters are set in pages, which are each 4K bytes.

total_log_used - Total Log Space Used

The total amount of active log space currently used (in bytes) in the database.

Table 1430. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Usage Use this element in conjunction with total_log_available to determine whether you may need to adjust the following configuration parameters to avoid running out of log space:

- logfilsiz
- logprimary
- logsecond

Note: While the database system monitor information is given in bytes, the configuration parameters are set in pages, which are each 4K bytes.

total_move_time - Total extent move time monitor element

In milliseconds, the total move time for all extents moved during the table space rebalance process.

Table 1431. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_EXTENT_MOVEMENT_STATUS - Get extent movement progress status metrics	Always collected

total_olap_funcs - Total OLAP Functions monitor element

The total number of OLAP functions executed.

Table 1432. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 1433. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage

At the database or application level, use this value in conjunction with olap_func_overflows to determine if a significant percentage of OLAP functions would benefit from modest increases in the sort heap size.

total_reorg_proc_time - Total reorganization processing time monitor element

The total amount of processing (non-wait) time spent performing reorg operations on the database server. The value is given in milliseconds.

Table 1434. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1435. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

total_reorg_time - Total reorganization time monitor element

The total amount of time spent performing reorg operations on the database server. The value is given in milliseconds.

Table 1436. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1437. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

total_reorgs - Total reorganizations monitor element

The number of reorg operations issued against the database server.

Table 1438. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1439. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

total_rollback_proc_time - Total rollback processing time monitor element

The total amount of processing (non-wait) time spent performing rollback operations on the database server. The value is given in milliseconds.

Table 1440. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1441. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

total_rollback_time - Total rollback time monitor element

The total amount of time spent performing rollback operations on the database server. The value is given in milliseconds.

Table 1442. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1443. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

total_routine_invocations - Total routine invocations monitor elements

The total number of times a routine was invoked.

Table 1444. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1445. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE

Table 1445. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

total_routine_non_sect_proc_time - Non-section processing time monitor element

The total amount of processing time this statement spent performing non-section execution within routines. This value includes both the time spent executing user-code within routines and time spent performing non-section operations like commit or rollback. Processing time does not include wait time. The value is given in milliseconds.

Table 1446. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 1447. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE

total_routine_non_sect_time - Non-section routine execution time monitor elements

The total amount of time this statement spent performing non-section execution within routines. This value includes both the time spent executing user-code within routines and the time spent performing non-section operations like commit or rollback. The value is given in milliseconds.

Table 1448. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 1449. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE

total_routine_time - Total routine time monitor element

The total time spent executing routines. The value is given in milliseconds.

Table 1450. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 1450. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Command and Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1451. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

If the collection level is set to BASE, then the value of **total_routine_time** monitor element does not include any time spent executing functions that were defined using the NO SQL clause.

If the collection level is set to EXTENDED, then the value of **total_routine_time** monitor element includes the time spent in all routines.

total_routine_user_code_proc_time - Total routine user code processing time monitor element

The total amount of processing time spent executing in routines outside of known DB2 times (typically user code in routines). The value is given in milliseconds.

Table 1452. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1453. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

If the collection level is set to BASE, then this monitor element does not include any processing time spent executing functions that were defined using the NO SQL clause. Instead, this time is included in the value of the **total_section_proc_time** monitor element.

If the collection level is set to EXTENDED, then the value of this monitor element includes the processing time spent executing all routines.

total_routine_user_code_time - Total routine user code time monitor element

The total amount of time spent executing in routines outside of known DB2 times (typically user code in routines). The value is given in milliseconds.

Table 1454. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1455. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE

Table 1455. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

If the collection level is set to BASE, then the value of this monitor element does not include the time spent executing functions that were defined using the NO SQL clause. Instead, this time is included in the value of the **total_section_time** monitor element.

If the collection level is set to EXTENDED, then the value of this monitor element includes the time spent executing all routines.

total_rqst_mapped_in - Total request mapped-in monitor element

The total number of requests that were mapped into this service subclass via a remap threshold or a work action set.

Table 1456. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1457. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_scmetrics	REQUEST METRICS BASE

total_rqst_mapped_out - Total request mapped-out monitor element

The total number of requests that were mapped out of this service subclass via a remap threshold or a work action set.

Table 1458. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 1458. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1459. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_scmetrics	REQUEST METRICS BASE

total_rqst_time - Total request time monitor element

The total amount of time spent working on requests. This value is reported in milliseconds.

Table 1460. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1461. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE

Table 1461. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

total_runstats - Total runtime statistics monitor element

The total number of runstats operations performed on the database server.

Table 1462. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1463. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

Table 1463. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

total_runstats_proc_time - Total runtime statistics processing time monitor element

The total amount of processing (non-wait) time spent performing runstats operations on the database server. The value is given in milliseconds. Any time the runstats utility spends throttled does not count to the runstats processing time.

Table 1464. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1465. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

Table 1465. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.		

total_runstats_time - Total runtime statistics time monitor element

The total amount of time spent performing runstats operations on the database server. The value is given in milliseconds.

Table 1466. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1467. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE
* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.		

total_sec_cons - Secondary Connections

The number of connections made by a subagent to the database at the node.

Element identifier
total_sec_cons

Element type
counter

Table 1468. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Usage You can use this element in conjunction with the total_cons, db_conn_time, and the db2start_time monitor elements to calculate the frequency at which applications have connected to the database.

total_section_proc_time - Total section processing time monitor element

The total amount of processing time agents spent performing section execution. Processing time does not include wait time. The value is given in milliseconds.

Table 1469. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1469. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Command and Level
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1470. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

If the collection level is set to BASE, then the value of the **total_section_proc_time** monitor element includes processing time spent executing functions that were defined using the NO SQL clause.

If the collection level is set to EXTENDED, then the processing time spent executing these functions is not included in the value of the **total_section_proc_time** monitor element. It is included in the value of the **total_routine_user_code_proc_time** monitor element.

total_section_sort_proc_time - Total section sort processing time monitor element

Total amount of processing (non-wait) time spent performing sorts while executing a section, which is the execution of the compiled query plan generated by the SQL statement that was issued by the client application. The value is given in milliseconds.

Table 1471. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 1472. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

Table 1472. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

At the system level, use this element with the **total_section_sorts** monitor element to calculate the average sort processing time (does not include waits) during section execution, which can indicate whether or not sorting is an issue as far as performance is concerned.

At the activity level, use this element to identify statements that spend a large amount of time sorting. These statements may benefit from additional tuning to reduce the sort time.

total_section_sort_time - Total section sort time monitor element

Total amount of time spent performing sorts while executing a section, which is the execution of the compiled query plan generated by the SQL statement that was issued by the client application. The value is given in milliseconds.

Table 1473. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

Table 1473. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 1474. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

At the system level, use this element with the **total_section_sorts** monitor element to calculate the average sort time during section execution, which can indicate whether or not sorting is an issue as far as statement performance is concerned.

The **total_section_sort_time** element includes both wait and processing time. If the value of (`total_section_sort_time - total_section_sort_proc_time`) is high, sorts are spending a lot of time waiting. For example, if sorts are frequently spilling to disk, the value of the **total_section_sort_time** monitor element will increase due to I/O waits. This time will not be included in the **total_section_sort_proc_time** monitor element value, which only counts the time actively processing a sort. In this case, you may consider tuning sort memory to improve performance.

At the activity level, use this element to identify statements that spend a large amount of time sorting. These statements may benefit from additional tuning to reduce the sort time.

total_section_sorts - Total section sorts monitor element

Total number of sorts performed during section execution, which is the execution of the compiled query plan generated by the SQL statement that was issued by the client application.

Table 1475. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 1476. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE

Table 1476. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Use this element with the**total_section_sort_time** monitor element to calculate the average amount of time spent performing sorts during section execution.

At the activity and package cache levels, use this element to identify statements which are performing large numbers of sorts. These statements may benefit from additional tuning to reduce the number of sorts. You can also use the EXPLAIN statement to identify the number of sorts a statement performs.

total_section_time - Total section time monitor element

The total time agents spent performing section execution. The value is given in milliseconds.

Table 1477. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 1477. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Command and Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1478. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

If the collection level is set to BASE, then the value of **total_section_time** monitor element includes time spent executing functions that were defined using the NO SQL clause.

If the collection level is set to EXTENDED, then the time spent executing these functions is not included in the value of the **total_section_time** monitor element. It is included in the value of the **total_routine_user_code_time** monitor element instead.

total_sort_time - Total sort time monitor element

The total elapsed time for all sorts that have been executed. This value is reported in milliseconds.

Table 1479. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Sort
Application	appl	Sort
Application	stmt	Sort
Dynamic SQL	dynsql	Sort

For snapshot monitoring, this counter can be reset.

Table 1480. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-
Statements	event_stmt	-
Activities	event_activity	Statement, Sort

Usage

At a database or application level, use this element with **total_sorts** to calculate the average sort time, which can indicate whether or not sorting is an issue as far as performance is concerned.

At a statement level, use this element to identify statements that spend a lot of time sorting. These statements may benefit from additional tuning to reduce the sort time.

This count also includes sort time of temporary tables created during related operations. It provides information for one statement, one application, or all applications accessing one database.

When using monitor elements providing elapsed times, you should consider:

1. Elapsed times are affected by system load, so the more processes you have running, the higher this elapsed time value.
2. To calculate this monitor element at a database level, the database system monitor sums the application-level times. This can result in double counting elapsed times at a database level, since more than one application process can be running at the same time.

   To provide meaningful data from the database level, you should normalize the data to a lower level. For example:

   ```
   total_sort_time / total_sorts
   ```

   provides information about the average elapsed time for each sort.

total_sorts - Total sorts monitor element

The total number of sorts that have been executed.

Table 1481. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 1482. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 1483. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE

Table 1483. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Database	event_db	-
Connection	event_conn	-
Statements	event_stmt	-
Activities	event_activity	Statement, Sort
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

At a database or application level, use this value with **sort_overflows** to calculate the percentage of sorts that need more heap space. You can also use it with **total_sort_time** to calculate the average sort time.

If the number of sort overflows is small with respect to the total sorts, then increasing the sort heap size may have little impact on performance, unless this buffer size is increased substantially.

At a statement level, use this element to identify statements which are performing large numbers of sorts. These statements may benefit from additional tuning to reduce the number of sorts. You can also use the SQL EXPLAIN statement to identify the number of sorts a statement performs.

total_sys_cpu_time - Total system CPU time for a statement monitor element

The total system CPU time for an SQL statement.

Table 1484. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Dynamic SQL	dynsql	Statement

For snapshot monitoring, this counter can be reset.

Usage

Use this element with Elapsed Statement Execution Time and Total User CPU for a Statement to evaluate which statements are the most expensive.

This element is composed of two subelements that report time spent as seconds and microseconds (one millionth of a second). The names of the subelements can be derived by adding "_s" and "_ms" to the name of this monitor element. To retrieve the total time spent for this monitor element, the values of the two subelements must be added together. For example, if the "_s" subelement value is 3 and the "_ms" subelement value is 20, then the total time spent for the monitor element is 3.00002 seconds.

total_usr_cpu_time - Total user CPU time for a statement monitor element

The total user CPU time for an SQL statement.

Table 1485. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Dynamic SQL	dynsql	Statement

For snapshot monitoring, this counter can be reset.

Usage

Use this element with Elapsed Statement Execution Time and to evaluate the longest running statements.

This element is composed of two subelements that report time spent as seconds and microseconds (one millionth of a second). The names of the subelements can be derived by adding "_s" and "_ms" to the name of this monitor element. To retrieve the total time spent for this monitor element, the values of the two subelements must be added together. For example, if the "_s" subelement value is 3 and the "_ms" subelement value is 20, then the total time spent for the monitor element is 3.00002 seconds.

total_wait_time - Total wait time monitor element

The total time spent waiting within the DB2 database server. The value is given in milliseconds.

Table 1486. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 1486. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1487. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

To understand the percentage of time the database server spends actively working on requests, use the following ratio:

```
(total_rqst_time - total_wait_time) / total_rqst_time
```

The value of the **client_idle_wait_time** monitor element is not included in the value of the **total_wait_time** monitor element. The **total_wait_time** element represents only time spent waiting while the database server is processing requests.

tpmon_acc_str - TP monitor client accounting string monitor element

The data passed to the target database for logging and diagnostic purposes, if the sqleseti API was issued in this connection. The current value of the CLIENT_ACCTNG special register for this connection, unit of work, or activity.

This monitor element is synonymous to the **client_acctng** monitor element. The **client_acctng** monitor element is used for monitoring table functions and event monitors that write to unformatted tables, which were introduced in DB2 Version 9.7. The **tpmon_acc_str** monitor element is used for snapshot monitors and event monitors that write to tables, files, and pipes.

Table 1488. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
DCS Application	dcs_appl	Basic

Table 1489. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-
Deadlock	event_dlconn	-
Transaction	event_xact	-

Usage

Use this element for problem determination and accounting purposes.

tpmon_client_app - TP monitor client application name monitor element

Identifies the server transaction program performing the transaction, if the sqleseti API was issued in this connection. The current value of the CLIENT_APPLNAME special register for this connection, unit of work, or activity.

This monitor element is synonymous to the **client_applname** monitor element. The **client_applname** monitor element is used for monitoring table functions and event monitors that write to unformatted tables, which were introduced in DB2 Version 9.7. The **tpmon_client_app** monitor element is used for snapshot monitors and event monitors that write to tables, files, and pipes.

Table 1490. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
DCS Application	dcs_appl	Basic

Table 1491. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-
Deadlock	event_dlconn	-
Transaction	event_xact	-

Usage

Use this element for problem determination and accounting purposes.

tpmon_client_userid - TP monitor client user ID monitor element

The client user ID generated by a transaction manager and provided to the server, if the sqleseti API is used. The current value of the CLIENT_USERID special register for this connection, unit of work, or activity.

This monitor element is synonymous to the **client_userid** monitor element. The **client_userid** monitor element is used for monitoring table functions and event monitors that write to unformatted tables, which were introduced in DB2 Version 9.7. The **tpmon_client_userid** monitor element is used for snapshot monitors and event monitors that write to tables, files, and pipes.

Table 1492. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
DCS Application	dcs_appl	Basic

Table 1493. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-
Deadlock	event_dlconn	-
Transaction	event_xact	-

Usage

Use this element in application server or Transaction Processing monitor environments to identify the end-user for whom the transaction is being executed.

tpmon_client_wkstn - TP monitor client workstation name monitor element

Identifies the client's system or workstation (for example CICS EITERMID), if the sqleseti API was issued in this connection. The current value of the CLIENT_WRKSTNNAME special register for this connection, unit of work, or activity.

This monitor element is synonymous to the **client_wrkstnname** monitor element. The **client_wrkstnname** monitor element is used for monitoring table functions and event monitors that write to unformatted tables, which were introduced in DB2 Version 9.7. The **tpmon_client_wkstn** monitor element is used for snapshot monitors and event monitors that write to tables, files, and pipes.

Table 1494. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
DCS Application	dcs_appl	Basic

Table 1495. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-
Deadlock	event_dlconn	-
Transaction	event_xact	-

Usage

Use this element to identify the user's machine by node ID, terminal ID, or similar identifiers.

tq_cur_send_spills - Current number of table queue buffers overflowed monitor element

The current number of table queue buffers residing in a temporary table.

Table 1496. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

Usage An agent writing to a table queue may be sending rows to several readers. The writing agent will overflow buffers to a temporary table when the agent that it is currently sending rows to is not accepting rows and another agent requires rows in order to proceed. Overflowing to temporary table allows both the writer and the other readers to continue processing.

Rows that have been overflowed will be sent to the reading agent when it is ready to accept more rows.

If this number is high, and queries fail with sqlcode -968, and there are messages in `db2diad.log` indicating that you ran out of temporary space in the TEMP table space, then table queue overflows may be the cause. This could indicate a problem on another node (such as locking). You would investigate by taking snapshots on all the partitions for this query.

There are also cases, perhaps because of the way data is partitioned, where many buffers need to be overflowed for the query. In these cases you will need to add more disk to the temporary table space.

tq_id_waiting_on - Waited on node on a table queue monitor element

The identifier of the table queue that is waiting to send or receive data.

Table 1497. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

Usage

This can be used for troubleshooting.

tq_max_send_spills - Maximum number of table queue buffers overflows

Maximum number of table queue buffers overflowed to a temporary table.

Element identifier
tq_max_send_spills

Element type
watermark

Table 1498. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

Table 1499. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_subsection	-

Usage Indicates the maximum number of table queue buffers that have been written to a temporary table.

tq_node_waited_for - Waited for node on a table queue

If the subsection status ss_status is *waiting to receive* or *waiting to send* and tq_wait_for_any is FALSE, then this is the number of the node that this agent is waiting for.

Element identifier
tq_node_waited_for

Element type
information

Table 1500. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

Usage This can be used for troubleshooting. You may want to take an application snapshot on the node that the subsection is waiting for. For example, the application could be in a lock wait on that node.

tq_rows_read - Number of Rows Read from table queues

Total number of rows read from table queues.

Element identifier
tq_rows_read

Element type
counter

Table 1501. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

Table 1502. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_subsection	-

Usage If monitoring does not indicate that this number is increasing, then processing progress is not taking place.

If there is significant differences in this number between nodes, then some nodes may be over utilized while others are being under utilized.

If this number is large, then there is a lot of data being shipped between nodes, suggest that optimization might improve the access plan.

tq_rows_written - Number of rows written to table queues

Total number of rows written to table queues.

Element identifier
tq_rows_written

Element type
counter

Table 1503. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

Table 1504. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_subsection	-

Usage If monitoring does not indicate that this number is increasing, then processing progress is not taking place.

If there is significant differences in this number between nodes, then some nodes may be over utilized while others are being under utilized.

If this number is large, then there is a lot of data being shipped between nodes, suggest that optimization might improve the access plan.

tq_tot_send_spills - Total number of table queue buffers overflowed monitor element

Total number of table queue buffers overflowed to a temporary table.

Table 1505. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 1506. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

Table 1507. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statements	event_subsection	-
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

Usage

Indicates the total number of table queue buffers that have been written to a temporary table. See the **tq_cur_send_spills** monitor element for more information.

tq_wait_for_any - Waiting for any node to send on a table queue

This flag is used to indicate that the subsection is blocked because it is waiting to receive rows from any node.

Element identifier
tq_wait_for_any

Element type
information

Table 1508. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

Usage If ss_status indicates *waiting to receive data on a table queue* and this flag is TRUE, then the subsection is waiting to receive rows from any node. This generally indicates that the SQL statement has not processed to the point it can pass data to the waiting agent. For example, the writing agent may be performing a sort and will not write rows until the sort has completed. From the db2expln output, determine the subsection number associated

with the tablequeue that the agent is waiting to receive rows from. You can then examine the status of that subsection by taking a snapshot on each node where it is executing.

ts_name - Table space being rolled forward monitor element

The name of the table space currently rolled forward.

Element identifier
ts_name

Element type
information

Table 1509. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	rollforward	Basic

Usage If a rollforward is in progress, this element identifies the table spaces involved.

uid_sql_stmts - Update/Insert/Delete SQL Statements Executed

The number of SQL UPDATE, INSERT, and DELETE statements that were executed.

Table 1510. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 1511. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage You can use this element to determine the level of database activity at the application or database level.

You can also use the following formula to determine the ratio of UPDATE, INSERT and DELETE statements to the total number of statements:

```
  uid_sql_stmts
/ (static_sql_stmts + dynamic_sql_stmts )
```

This information can be useful for analyzing application activity and throughput.

unread_prefetch_pages - Unread prefetch pages monitor element

Indicates the number of pages that the prefetcher read in that were never used.

Table 1512. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1513. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 1514. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-

Usage

If this number is high, prefetchers are causing unnecessary I/O by reading pages into the buffer pool that will not be used.

uow_comp_status - Unit of Work Completion Status

The status of the unit of work and how it stopped.

Element identifier
uow_comp_status

Element type
information

Table 1515. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Unit of Work
DCS Application	dcs_appl	Basic

Table 1516. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Transactions	event_xact	-

Usage You may use this element to determine if the unit of work ended due to a deadlock or abnormal termination. It may have been:

- Committed due to a commit statement
- Rolled back due to a rollback statement
- Rolled back due to a deadlock
- Rolled back due to an abnormal termination
- Committed at normal application termination.
- Unknown as a result of a FLUSH EVENT MONITOR command for which units of work were in progress.

Note: API users should refer to the header file (*sqlmon.h*) containing definitions of database system monitor constants.

uow_elapsed_time - Most Recent Unit of Work Elapsed Time

The elapsed execution time of the most recently completed unit of work.

Element identifier
uow_elapsed_time

Element type
time

Table 1517. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Unit of Work, Timestamp
DCS Application	dcs_appl	Unit of Work, Timestamp

Usage

Use this element as an indicator of the time it takes for units of work to complete.

This element is composed of two subelements that report time spent as seconds and microseconds (one millionth of a second). The names of the subelements can be derived by adding "_s" and "_ms" to the name of this monitor element. To retrieve the total time spent for this monitor element, the values of the two subelements must be added together. For example, if the "_s" subelement value is 3 and the "_ms" subelement value is 20, then the total time spent for the monitor element is 3.00002 seconds.

uow_id - Unit of work ID monitor element

The unit of work identifier. The unit of work ID is unique within an application handle.

Table 1518. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics	Always collected

Table 1518. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details	Always collected

Table 1519. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-
Activities	event_activity	-
Activities	event_activitystmt	-
Activities	event_activityvals	-
Threshold violations	event_thresholdviolations	-

Usage

Use this element in conjunction with other activity history elements for analysis of the behavior of an activity.

You can also use this element with the **activity_id** and **appl_id** monitor elements to uniquely identify an activity.

uow_lock_wait_time - Total time unit of work waited on locks monitor element

The total amount of elapsed time this unit of work has spent waiting for locks. The value is given in milliseconds.

Element identifier
uow_lock_wait_time

Element type
counter

Table 1520. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Unit of Work

Usage This element can help you determine the severity of the resource contention problem.

uow_log_space_used - Unit of work log space used monitor element

The amount of log space (in bytes) used in the current unit of work of the monitored application.

Table 1521. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected

Table 1521. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected

Table 1522. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Unit of Work

Table 1523. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Transactions	event_xact	-
Unit of work	-	-

Usage

You may use this element to understand the logging requirements at the unit of work level.

uow_start_time - Unit of work start timestamp monitor element

The date and time that the unit of work first required database resources.

Table 1524. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected

Table 1525. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Unit of Work, Timestamp
DCS Application	dcs_appl	Unit of Work, Timestamp

Table 1526. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-
Transaction	event_xact	-

Usage

This resource requirement occurs at the first SQL statement execution of that unit of work:

- For the first unit of work, it is the time of the first database request (SQL statement execution) after **conn_complete_time**.
- For subsequent units of work, it is the time of the first database request (SQL statement execution) after the previous COMMIT or ROLLBACK.

Note: The *SQL Reference* defines the boundaries of a unit of work as the COMMIT or ROLLBACK points.

The database system monitor excludes the time spent between the COMMIT/ROLLBACK and the next SQL statement from its definition of a unit of work. This measurement method reflects the time spent by the database manager in processing database requests, separate from time spent in application logic before the first SQL statement of that unit of work. The unit of work elapsed time does include the time spent running application logic between SQL statements within the unit of work.

You may use this element with the **uow_stop_time** monitor element to calculate the total elapsed time of the unit of work and with the **prev_uow_stop_time** monitor element to calculate the time spent in the application between units of work.

You can use the **uow_stop_time** and the **prev_uow_stop_time** monitor elements to calculate the elapsed time for the *SQL Reference* definition of a unit of work.

uow_status - Unit of Work Status

The status of the unit of work.

Element identifier
uow_status

Element type
information

Table 1527. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Transactions	event_xact	-

Usage You may use this element to determine the status of a unit of work. API users should refer to the sqlmon.h header file containing definitions of database system monitor constants.

uow_stop_time - Unit of work stop timestamp monitor element

The date and time that the most recent unit of work completed, which occurs when database changes are committed or rolled back.

Table 1528. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Unit of Work, Timestamp
DCS Application	dcs_appl	Unit of Work, Timestamp

Table 1529. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-

Usage

Use this element with the **prev_uow_stop_time** monitor element to calculate the total elapsed time between COMMIT/ROLLBACK points, and with the **uow_start_time** monitor element to calculate the elapsed time of the latest unit of work.

The timestamp contents will be set as follows:

- When the application has completed a unit of work and has not yet started a new one (as defined in the **uow_start_time** monitor element), this element reports a valid, non-zero timestamp.
- When the application is currently executing a unit of work, this element reports zeros.
- When the application first connects to the database, this element is set to the value of the **conn_complete_time** monitor element

As a new unit of work is started, the contents of this element are moved to the **prev_uow_stop_time** monitor element.

uow_total_time_top - UOW total time top monitor element

High watermark for unit of work lifetime, in milliseconds.

Table 1530. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wlstats	-
Statistics	event_scstats	-

Usage

This element can be used to help determine whether or not the UOWTOTALTIME threshold is effective and can also help to determine how to configure such a threshold.

For service classes, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service class is set to NONE.

For workloads, this monitor element returns -1 if COLLECT AGGREGATE ACTIVITY DATA for the workload is set to NONE.

For a service class, measurements taken for this high watermark are computed for the service class assigned by the workload. Any mapping by a work action set to change the service class of an activity does not affect this high watermark.

update_sql_stmts - Updates

This element contains a count of the total number of times the federated server has issued an UPDATE statement to this data source on behalf of any application from the start of the federated server instance, or the last reset of the database monitor counters.

Table 1531. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

Usage Use this element to determine the level of database activity directed against this data source by the federated server or an application.

You can also use this element to determine the percentage of write activity against this data source by the federated server or an application, with the following formula:

```
write_activity =
  (INSERT statements + UPDATE statements + DELETE statements ) /
  (SELECT statements + INSERT statements + UPDATE statements +
  DELETE statements)
```

update_time - Update Response Time

This element contains the aggregate amount of time, in milliseconds, that it has taken this data source to respond to UPDATEs from all applications or a single application running on this federated server instance from the start of the federated server instance, or the last reset of the database monitor counters.

Table 1532. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Timestamp
Application	appl_remote	Timestamp

For snapshot monitoring, this counter can be reset.

The response time is measured as the difference in time between the time the federated server submits an UPDATE statement to the data source, and the time the data source responds to the federated server, indicating the UPDATE has been processed.

Usage Use this element to determine how much actual time transpires while waiting for UPDATEs to this data source to be processed. This information can be useful for capacity planning and tuning.

user_cpu_time - User CPU time monitor element

The total *user* CPU time (in seconds and microseconds) used by the database manager agent process, the unit of work, or the statement. For event monitors that write to tables, the value of this element is given in microseconds by using the BIGINT data type.

When either the statement monitor switch or the timestamp switch is not turned on, this element is not collected and -1 is written instead.

Table 1533. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

Table 1533. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Transactions	event_xact	-
Statements	event_stmt	-
Activities	event_activity	-

Usage

This element along with the other related CPU-time elements can help you understand the level of activity within an application, and can help you identify applications that could benefit from additional tuning.

Note: If this information is not available for your operating system, this element will be set to 0.

Note: Due to the differences in granularity with which the DB2 system collects statistics, the value of the **total_exec_time** monitor element might not equal the sum of values of **system_cpu_time** and **user_cpu_time** monitor elements. In this case, the sum of **system_cpu_time** and **user_cpu_time** monitor elements more accurately reflects the actual total execution time.

utility_dbname - Database Operated on by Utility

The database operated on by the utility.

Table 1534. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	utility_info	Basic

utility_description - Utility Description

A brief description of the work a utility is performing. For example, a rebalance invocation may contain "Tablespace ID: 2" representing that this rebalancer is working on table space with ID 2. The format of this field is dependent on the class of utility and is subject to change between releases.

Table 1535. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	utility_info	Basic

utility_id - Utility ID

The unique identifier corresponding to the utility invocation.

Table 1536. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 1537. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	utility_info	Basic

utility_invoker_type - Utility Invoker Type

This element describes how a utility was invoked.

Table 1538. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	utility_info	Basic

Usage Use this element to determine how a utility was invoked. For example, you can use it to determine whether a utility was invoked automatically by DB2 or by a user. The values for this element, listed as follows, are defined in sqlmon.h.

API Constant	Utility
SQLM_UTILITY_INVOKER_USER	Utility was invoked by user
SQLM_UTILITY_INVOKER_AUTO	Utility was invoked automatically by DB2

utility_priority - Utility Priority

Utility priority specifies the amount of relative importance of a throttled utility with respect to its throttled peers. A priority of 0 implies that a utility is executing unthrottled. Non-zero priorities must fall in the range of 1-100, with 100 representing the highest priority and 1 representing the lowest.

Table 1539. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	utility_info	Basic

utility_start_time - Utility Start Time

The date and time when the current utility was originally invoked.

Table 1540. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	utility_info	Basic

utility_state - Utility State

This element describes the state of a utility.

Element identifier
utility_state

Element type
information

Table 1541. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	utility_info	Basic

Usage Use this element to determine the state of an active utility. The values for this field, listed as follows, are defined in sqlmon.h.

API Constant	Description
SQLM_UTILITY_STATE_EXECUTE	Utility is executing
SQLM_UTILITY_STATE_WAIT	Utility is waiting for an event to occur before resuming progress
SQLM_UTILITY_STATE_ERROR	Utility has encountered an error

utility_type - Utility Type

The class of utility.

Table 1542. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	utility_info	Basic

Usage

The values for this element can be any of the constants defined in `sqlmon.h` with names beginning "SQLM_UTILITY_".

valid - Section validity indicator monitor element

Indicates whether the dynamic SQL statement section is valid. For static SQL statements, the value of this monitor element is always Y.

Table 1543. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	Always collected

Usage

Valid values for this monitor element are Y and N. An invalid section will be implicitly prepared by the system when next used.

vectored_ios - Number of vectored I/O requests monitor element

The number of vectored I/O requests. More specifically, the number of times DB2 performs sequential prefetching of pages into the page area of the buffer pool.

Table 1544. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE

Table 1545. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool	Buffer Pool

Usage

Use this element to determine how often vectored I/O is being done. The number of vectored I/O requests is monitored only during sequential prefetching.

version - Version of Monitor Data

The version of the database manager that produced the event monitor data stream.

Table 1546. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Event Log Header	event_log_header	-

Usage

The data structures used by the event monitor may change between releases of the database manager. As a result, your monitor applications should check the version of the data stream to determine if they can process the data they will be receiving.

For this release, this element is set to the API constant SQLM_DBMON_VERSION9_5.

virtual_mem_free - Free virtual memory monitor element

The amount of virtual memory available on this host that is not allocated to any process, in MB.

Table 1547. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

virtual_mem_reserved - Reserved virtual memory monitor element

The amount of virtual memory reserved by running processes, in MB.

Table 1548. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

virtual_mem_total - Total virtual memory monitor element

The total amount of virtual memory available on this host, in MB.

Table 1549. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
ENV_GET_SYSTEM_RESOURCES table function - Return system information	Always collected

wl_work_action_set_id - Workload work action set identifier monitor element

If this activity has been categorized into a work class of workload scope, this monitor element shows the ID of the work action set associated with the work class set to which the work class belongs. Otherwise, this monitor element shows the value of `0`.

Table 1550. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 1551. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

Usage

Use this monitor element, together with the **wl_work_class_id** monitor element, to uniquely identify the workload work class of the activity, if one exists.

wl_work_class_id - Workload work class identifier monitor element

If this activity has been categorized into a work class of workload scope, then this monitor element displays the identifier of the work class. Otherwise, this monitor element displays the value of 0.

Table 1552. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 1553. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

Usage

Use this monitor element, together with the **wl_work_action_set_id** monitor element, to uniquely identify the workload work class of the activity, if one exists.

wlm_queue_assignments_total - Workload manager total queue assignments monitor element

The number of times that activities have been queued by a WLM threshold.

Table 1554. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1554. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 1555. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

wlm_queue_time_total - Workload manager total queue time monitor element

The time spent waiting on a WLM queuing threshold. This value is given in milliseconds.

Table 1556. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 1556. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS table function - get detailed metrics for package cache entries	ACTIVITY METRICS BASE

Table 1557. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Activities	event_activitymetrics	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Package cache	Reported in the activity_metrics document.	ACTIVITY METRICS BASE
Unit of work	Reported in the system_metrics document.	REQUEST METRICS BASE
Statistics	event_scmetrics*	REQUEST METRICS BASE
Statistics	event_wlmetrics*	REQUEST METRICS BASE

* When returned as part of this logical data group, this element reflects the *change in value* of this metric since the last statistics collection or database activation, whichever was more recent.

wlo_completed_total - Workload occurrences completed total monitor element

The number of workload occurrences to complete since last reset.

Table 1558. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wlstats	-

Usage

Use this element to determine how many occurrences of a given workload are driving work into the system.

work_action_set_id - Work action set ID monitor element

The ID of the work action set to which this statistics record applies.

Table 1559. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_histogrambin	-
Statistics	event_wcstats	-

Usage

Use this element in conjunction with other activity history elements for analysis of the behavior of an activity or with other statistics elements for analysis of a work class.

The value of this element is 0 when the following conditions are met:

- The element is reported in an event_histogrambin logical data group.
- The histogram data is collected for an object that is not a work class.

work_action_set_name - Work action set name monitor element

The name of the work action set to which the statistics shown as part of this event are associated.

Table 1560. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_qstats	-
Statistics	event_wcstats	-

Usage

Use this element along with the **work_class_name** element to uniquely identify the work class whose statistics are being shown in this record or to uniquely identify the work class which is the domain of the threshold queue whose statistics are shown in this record.

work_class_id - Work class ID monitor element

The identifier of the work class to which this statistics record applies.

Table 1561. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wcstats	-
Statistics	event_histogrambin	-

Usage

Use this element in conjunction with other statistics elements for analysis of a work class.

The value of this element is 0 when the following conditions are met:

- The element is reported in an event_histogrambin logical data group.
- The histogram data is collected for an object that is not a work class.

work_class_name - Work class name monitor element

The name of the work class to which the statistics shown as part of this event are associated.

Table 1562. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_qstats	-
Statistics	event_wcstats	-

Usage

Use this element along with the **work_action_set_name** element to uniquely identify the work class whose statistics are being shown in this record or to uniquely identify the work class which is the domain of the threshold queue whose statistics are shown in this record.

workload_id - Workload ID monitor element

An integer that uniquely identifies a workload.

Table 1563. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_WORKLOAD table function - Get workload metrics	Always collected
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics	Always collected

Table 1564. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic

Table 1565. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Statistics	event_wlstats	-
Statistics	event_histogrambin	-
Activities	event_activity	-
Threshold violations	event_thresholdviolations	-

Usage

Use this ID to uniquely identify the workload to which this activity, application, histogram bin, or workload statistics record belongs.

The value of this element is 0 when the following conditions are met:

- The element is reported in an event_histogrambin logical data group.
- The histogram data is collected for an object that is not a workload.

workload_name - Workload name monitor element

Name of the workload.

Table 1566. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics	Always collected
MON_GET_WORKLOAD table function - Get workload metrics	Always collected
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics	Always collected

Table 1567. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE

Table 1567. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the metrics document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the metrics document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Statistics	event_wlstats	-

Usage

In the statistics event monitor and workload table functions, the workload name identifies the workload for which statistics or metrics are being collected and reported. In the unit of work event monitor and unit of work table functions, the workload name identifies the workload that the unit of work was associated with.

Use the workload name to identify units of work or sets of information that apply to a particular workload of interest.

workload_occurrence_id - Workload occurrence identifier monitor element

The ID of the workload occurrence to which this activity belongs.

Table 1568. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics	Always collected

Table 1569. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-
Activities	event_activity	-

Usage

Use this to identify the workload occurrence that submitted the activity.

workload_occurrence_state - Workload occurrence state monitor element

The state of the workload occurrence.

Table 1570. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected

Usage

Possible values include:

DECOUPLED
Workload occurrence does not have a coordinator agent assigned (concentrator case).

DISCONNECTPEND
Workload occurrence is disconnecting from the database.

FORCED
Workload occurrence has been forced.

INTERRUPTED
Workload occurrence has been interrupted.

QUEUED
Workload occurrence coordinator agent is queued by Query Patroller or a workload management queuing threshold. In a partitioned database environment, this state may indicate that the coordinator agent has made an RPC to the catalog partition to obtain threshold tickets and has not yet received a response.

TRANSIENT
Workload occurrence has not yet been mapped to a service superclass.

UOWEXEC
Workload occurrence is processing a request.

UOWWAIT
Workload occurrence is waiting for a request from the client.

x_lock_escals - Exclusive lock escalations monitor element

The number of times that locks have been escalated from several row locks to one exclusive table lock, or the number of times an exclusive lock on a row caused the table lock to become an exclusive lock.

Table 1571. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Table 1571. Snapshot Monitoring Information (continued)

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 1572. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-
Transactions	event_xact	-

Usage

Other applications cannot access data held by an exclusive lock; therefore it is important to track exclusive locks since they can impact the concurrency of your data.

A lock is escalated when the total number of locks held by an application reaches the maximum amount of lock list space available to the application. The amount of lock list space available is determined by the **locklist** and **maxlocks** configuration parameters.

When an application reaches the maximum number of locks allowed and there are no more locks to escalate, it will then use space in the lock list allocated for other applications. When the entire lock list is full, an error occurs.

See the **lock_escals** monitor element for possible causes and resolutions to excessive exclusive lock escalations.

An application may be using exclusive locks when share locks are sufficient. Although share locks may not reduce the total number of lock escalations share lock escalations may be preferable to exclusive lock escalations.

xda_object_pages - XDA Object Pages

The number of disk pages consumed by XML storage object (XDA) data.

Element identifier
xda_object_pages

Element type
information

Table 1573. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table	Basic

Table 1574. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Tables	event_table	-

Usage This element provides a mechanism for viewing the actual amount of space consumed by XML storage object (XDA) data in a particular table. This element can be used in conjunction with a table event monitor to track the rate of XML storage object data growth over time.

xda_object_l_pages - XML storage object (XDA) data logical pages monitor element

The number of logical pages used on disk by XML storage object (XDA) data.

Table 1575. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLE table function - Get table metrics	DATA OBJECT METRICS BASE

Usage

- This value might be less than the amount of space physically allocated for the object. This can happen when you use the REUSE STORAGE option of the TRUNCATE statement. This option causes storage allocated for the table to continue to be allocated, although the storage will be considered empty. In addition, the value for this monitor element might be less than the amount of space logically allocated for the object, because the total space logically allocated includes a small amount of additional meta data.

  To retrieve an accurate measure of the logical or physical size of an object, use the ADMIN_GET_TAB_INFO_V97 function. This function provides more accurate information about the size of objects than you can obtain by multiplying the number of pages reported for this monitor element by the page size.

xid - Transaction ID

A unique transaction identifier (across all databases) generated by a transaction manager in a two-phase commit transaction.

Table 1576. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcs_appl	Unit of Work

Usage This identifier can be used to correlate the transaction generated by the transaction manager with the transactions executed against multiple databases. It can be used to help diagnose transaction manager problems by tying database transactions involving a two-phase commit protocol with the transactions originated by the transaction manager.

xquery_stmts - XQuery Statements Attempted

The number of XQuery statements executed for an application or database.

Element identifier
xquery_stmts

Element type
counter

Table 1577. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 1578. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

Usage You can use this element to gauge the activity of native XQuery language requests. This does not include embedded XQuery language requests such as xmlquery, xmltable, or xmlexist.

Part 3. Appendixes

© Copyright IBM Corp. 1993, 2012

Appendix A. Overview of the DB2 technical information

DB2 technical information is available through the following tools and methods:

- DB2 Information Center
  - Topics (Task, concept and reference topics)
  - Help for DB2 tools
  - Sample programs
  - Tutorials
- DB2 books
  - PDF files (downloadable)
  - PDF files (from the DB2 PDF DVD)
  - printed books
- Command line help
  - Command help
  - Message help

Note: The DB2 Information Center topics are updated more frequently than either the PDF or the hardcopy books. To get the most current information, install the documentation updates as they become available, or refer to the DB2 Information Center at ibm.com.

You can access additional DB2 technical information such as technotes, white papers, and IBM Redbooks® publications online at ibm.com. Access the DB2 Information Management software library site at http://www.ibm.com/software/data/sw-library/.

Documentation feedback

We value your feedback on the DB2 documentation. If you have suggestions for how to improve the DB2 documentation, send an email to db2docs@ca.ibm.com. The DB2 documentation team reads all of your feedback, but cannot respond to you directly. Provide specific examples wherever possible so that we can better understand your concerns. If you are providing feedback on a specific topic or help file, include the topic title and URL.

Do not use this e-mail address to contact DB2 Customer Support. If you have a DB2 technical issue that the documentation does not resolve, contact your local IBM service center for assistance.

DB2 technical library in hardcopy or PDF format

The following tables describe the DB2 library available from the IBM Publications Center at www.ibm.com/e-business/linkweb/publications/servlet/pbi.wss. English Version 9.7 manuals in PDF format can be downloaded from www.ibm.com/support/docview.wss?uid=swg27015148 and translated DB2 manuals in PDF format can be downloaded from www.ibm.com/support/docview.wss?uid=swg27015149.

© Copyright IBM Corp. 1993, 2012

Although the tables identify books available in print, the books might not be available in your country or region.

The form number increases each time a manual is updated. Ensure that you are reading the most recent version of the manuals, as listed below.

Note: The *DB2 Information Center* is updated more frequently than either the PDF or the hard-copy books.

Table 1579. DB2 technical information

Name	Form Number	Available in print	Last updated
Administrative API Reference	SC27-2435-03	Yes	July, 2012
Administrative Routines and Views	SC27-2436-03	No	July, 2012
Call Level Interface Guide and Reference, Volume 1	SC27-2437-03	Yes	July, 2012
Call Level Interface Guide and Reference, Volume 2	SC27-2438-03	Yes	July, 2012
Command Reference	SC27-2439-03	Yes	July, 2012
Data Movement Utilities Guide and Reference	SC27-2440-01	Yes	July, 2012
Data Recovery and High Availability Guide and Reference	SC27-2441-03	Yes	July, 2012
Database Administration Concepts and Configuration Reference	SC27-2442-03	Yes	July, 2012
Database Monitoring Guide and Reference	SC27-2458-03	Yes	July, 2012
Database Security Guide	SC27-2443-02	Yes	July, 2012
DB2 Text Search Guide	SC27-2459-03	Yes	July, 2012
Developing ADO.NET and OLE DB Applications	SC27-2444-02	Yes	July, 2012
Developing Embedded SQL Applications	SC27-2445-02	Yes	July, 2012
Developing Java Applications	SC27-2446-03	Yes	July, 2012
Developing Perl, PHP, Python, and Ruby on Rails Applications	SC27-2447-02	No	July, 2012
Developing User-defined Routines (SQL and External)	SC27-2448-02	Yes	July, 2012
Getting Started with Database Application Development	GI11-9410-02	Yes	July, 2012

Table 1579. DB2 technical information (continued)

Name	Form Number	Available in print	Last updated
Getting Started with DB2 Installation and Administration on Linux and Windows	GI11-9411-00	Yes	August, 2009
Globalization Guide	SC27-2449-00	Yes	August, 2009
Installing DB2 Servers	GC27-2455-03	Yes	July, 2012
Installing IBM Data Server Clients	GC27-2454-02	No	July, 2012
Message Reference Volume 1	SC27-2450-01	No	August, 2009
Message Reference Volume 2	SC27-2451-01	No	August, 2009
Net Search Extender Administration and User's Guide	SC27-2469-02	No	September, 2010
Partitioning and Clustering Guide	SC27-2453-02	Yes	July, 2012
pureXML Guide	SC27-2465-02	Yes	July, 2012
Query Patroller Administration and User's Guide	SC27-2467-00	No	August, 2009
Spatial Extender and Geodetic Data Management Feature User's Guide and Reference	SC27-2468-02	No	July, 2012
SQL Procedural Languages: Application Enablement and Support	SC27-2470-03	Yes	July, 2012
SQL Reference, Volume 1	SC27-2456-03	Yes	July, 2012
SQL Reference, Volume 2	SC27-2457-03	Yes	July, 2012
Troubleshooting and Tuning Database Performance	SC27-2461-03	Yes	July, 2012
Upgrading to DB2 Version 9.7	SC27-2452-03	Yes	July, 2012
Visual Explain Tutorial	SC27-2462-00	No	August, 2009
What's New for DB2 Version 9.7	SC27-2463-03	Yes	July, 2012
Workload Manager Guide and Reference	SC27-2464-03	Yes	July, 2012
XQuery Reference	SC27-2466-01	No	November, 2009

Table 1580. DB2 Connect-specific technical information

Name	Form Number	Available in print	Last updated
Installing and Configuring DB2 Connect Personal Edition	SC27-2432-03	Yes	July, 2012
Installing and Configuring DB2 Connect Servers	SC27-2433-03	Yes	July, 2012
DB2 Connect User's Guide	SC27-2434-02	Yes	September, 2010

Table 1581. Information Integration technical information

Name	Form Number	Available in print	Last updated
Information Integration: Administration Guide for Federated Systems	SC19-1020-02	Yes	August, 2009
Information Integration: ASNCLP Program Reference for Replication and Event Publishing	SC19-1018-04	Yes	August, 2009
Information Integration: Configuration Guide for Federated Data Sources	SC19-1034-02	No	August, 2009
Information Integration: SQL Replication Guide and Reference	SC19-1030-02	Yes	August, 2009
Information Integration: Introduction to Replication and Event Publishing	GC19-1028-02	Yes	August, 2009

Ordering printed DB2 books

About this task

If you require printed DB2 books, you can buy them online in many but not all countries or regions. You can always order printed DB2 books from your local IBM representative. Keep in mind that some softcopy books on the *DB2 PDF Documentation* DVD are unavailable in print. For example, neither volume of the *DB2 Message Reference* is available as a printed book.

Printed versions of many of the DB2 books available on the DB2 PDF Documentation DVD can be ordered for a fee from IBM. Depending on where you are placing your order from, you may be able to order books online, from the IBM Publications Center. If online ordering is not available in your country or region, you can always order printed DB2 books from your local IBM representative. Note that not all books on the DB2 PDF Documentation DVD are available in print.

Note: The most up-to-date and complete DB2 documentation is maintained in the DB2 Information Center at http://publib.boulder.ibm.com/infocenter/db2luw/v9r7.

To order printed DB2 books:

Procedure

- To find out whether you can order printed DB2 books online in your country or region, check the IBM Publications Center at http://www.ibm.com/shop/publications/order. You must select a country, region, or language to access publication ordering information and then follow the ordering instructions for your location.
- To order printed DB2 books from your local IBM representative:
  1. Locate the contact information for your local representative from one of the following websites:
     - The IBM directory of world wide contacts at www.ibm.com/planetwide
     - The IBM Publications website at http://www.ibm.com/shop/publications/order. You will need to select your country, region, or language to the access appropriate publications home page for your location. From this page, follow the "About this site" link.
  2. When you call, specify that you want to order a DB2 publication.
  3. Provide your representative with the titles and form numbers of the books that you want to order. For titles and form numbers, see “DB2 technical library in hardcopy or PDF format” on page 1073.

Displaying SQL state help from the command line processor

DB2 products return an SQLSTATE value for conditions that can be the result of an SQL statement. SQLSTATE help explains the meanings of SQL states and SQL state class codes.

Procedure

To start SQL state help, open the command line processor and enter:

```
? sqlstate or ? class code
```

where *sqlstate* represents a valid five-digit SQL state and *class code* represents the first two digits of the SQL state.
For example, `? 08003` displays help for the 08003 SQL state, and `? 08` displays help for the 08 class code.

Accessing different versions of the DB2 Information Center

About this task

For DB2 Version 9.8 topics, the *DB2 Information Center* URL is http://publib.boulder.ibm.com/infocenter/db2luw/v9r8/.

For DB2 Version 9.7 topics, the *DB2 Information Center* URL is http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/.

For DB2 Version 9.5 topics, the *DB2 Information Center* URL is http://publib.boulder.ibm.com/infocenter/db2luw/v9r5.

For DB2 Version 9.1 topics, the *DB2 Information Center* URL is http://publib.boulder.ibm.com/infocenter/db2luw/v9/.

For DB2 Version 8 topics, go to the *DB2 Information Center* URL at: http://publib.boulder.ibm.com/infocenter/db2luw/v8/.

Displaying topics in your preferred language in the DB2 Information Center

About this task

The DB2 Information Center attempts to display topics in the language specified in your browser preferences. If a topic has not been translated into your preferred language, the DB2 Information Center displays the topic in English.

Procedure

- To display topics in your preferred language in the Internet Explorer browser:
  1. In Internet Explorer, click the **Tools** —> **Internet Options** —> **Languages...** button. The Language Preferences window opens.
  2. Ensure your preferred language is specified as the first entry in the list of languages.
     - To add a new language to the list, click the **Add...** button.

       Note: Adding a language does not guarantee that the computer has the fonts required to display the topics in the preferred language.
     - To move a language to the top of the list, select the language and click the **Move Up** button until the language is first in the list of languages.
  3. Refresh the page to display the DB2 Information Center in your preferred language.
- To display topics in your preferred language in a Firefox or Mozilla browser:
  1. Select the button in the **Languages** section of the **Tools** —> **Options** —> **Advanced** dialog. The Languages panel is displayed in the Preferences window.
  2. Ensure your preferred language is specified as the first entry in the list of languages.
     - To add a new language to the list, click the **Add...** button to select a language from the Add Languages window.
     - To move a language to the top of the list, select the language and click the **Move Up** button until the language is first in the list of languages.
  3. Refresh the page to display the DB2 Information Center in your preferred language.

Results

On some browser and operating system combinations, you must also change the regional settings of your operating system to the locale and language of your choice.

Updating the DB2 Information Center installed on your computer or intranet server

A locally installed DB2 Information Center must be updated periodically.

Before you begin

A DB2 Version 9.7 Information Center must already be installed. For details, see the "Installing the DB2 Information Center using the DB2 Setup wizard" topic in *Installing DB2 Servers*. All prerequisites and restrictions that applied to installing the Information Center also apply to updating the Information Center.

About this task

An existing DB2 Information Center can be updated automatically or manually:

- Automatic updates - updates existing Information Center features and languages. An additional benefit of automatic updates is that the Information Center is unavailable for a minimal period of time during the update. In addition, automatic updates can be set to run as part of other batch jobs that run periodically.
- Manual updates - should be used when you want to add features or languages during the update process. For example, a local Information Center was originally installed with both English and French languages, and now you want to also install the German language; a manual update will install German, as well as, update the existing Information Center features and languages. However, a manual update requires you to manually stop, update, and restart the Information Center. The Information Center is unavailable during the entire update process.

This topic details the process for automatic updates. For manual update instructions, see the "Manually updating the DB2 Information Center installed on your computer or intranet server" topic.

Procedure

To automatically update the DB2 Information Center installed on your computer or intranet server:

1. On Linux operating systems,
   a. Navigate to the path where the Information Center is installed. By default, the DB2 Information Center is installed in the `/opt/ibm/db2ic/V9.7` directory.
   b. Navigate from the installation directory to the `doc/bin` directory.
   c. Run the `update-ic` script:
      ```
      update-ic
      ```
2. On Windows operating systems,
   a. Open a command window.
   b. Navigate to the path where the Information Center is installed. By default, the DB2 Information Center is installed in the `<Program Files>\IBM\DB2 Information Center\Version 9.7` directory, where `<Program Files>` represents the location of the Program Files directory.
   c. Navigate from the installation directory to the `doc\bin` directory.
   d. Run the `update-ic.bat` file:
      ```
      update-ic.bat
      ```

Results

The DB2 Information Center restarts automatically. If updates were available, the Information Center displays the new and updated topics. If Information Center

updates were not available, a message is added to the log. The log file is located in `doc\eclipse\configuration` directory. The log file name is a randomly generated number. For example, `1239053440785.log`.

Manually updating the DB2 Information Center installed on your computer or intranet server

If you have installed the DB2 Information Center locally, you can obtain and install documentation updates from IBM.

About this task

Updating your locally-installed *DB2 Information Center* manually requires that you:

1. Stop the *DB2 Information Center* on your computer, and restart the Information Center in stand-alone mode. Running the Information Center in stand-alone mode prevents other users on your network from accessing the Information Center, and allows you to apply updates. The Workstation version of the DB2 Information Center always runs in stand-alone mode. .
2. Use the Update feature to see what updates are available. If there are updates that you must install, you can use the Update feature to obtain and install them

   Note: If your environment requires installing the *DB2 Information Center* updates on a machine that is not connected to the internet, mirror the update site to a local file system using a machine that is connected to the internet and has the *DB2 Information Center* installed. If many users on your network will be installing the documentation updates, you can reduce the time required for individuals to perform the updates by also mirroring the update site locally and creating a proxy for the update site.
   If update packages are available, use the Update feature to get the packages. However, the Update feature is only available in stand-alone mode.
3. Stop the stand-alone Information Center, and restart the *DB2 Information Center* on your computer.

Note: On Windows 2008, Windows Vista (and higher), the commands listed later in this section must be run as an administrator. To open a command prompt or graphical tool with full administrator privileges, right-click the shortcut and then select **Run as administrator**.

Procedure

To update the *DB2 Information Center* installed on your computer or intranet server:

1. Stop the *DB2 Information Center*.
   - On Windows, click **Start** > **Control Panel** > **Administrative Tools** > **Services**. Then right-click **DB2 Information Center** service and select **Stop**.
   - On Linux, enter the following command:
     ```
     /etc/init.d/db2icdv97 stop
     ```
2. Start the Information Center in stand-alone mode.
   - On Windows:
     a. Open a command window.
     b. Navigate to the path where the Information Center is installed. By default, the *DB2 Information Center* is installed in the

Program_Files\IBM\DB2 Information Center\Version 9.7 directory, where *Program_Files* represents the location of the Program Files directory.

c. Navigate from the installation directory to the doc\bin directory.

d. Run the help_start.bat file:

```
help_start.bat
```

- On Linux:

a. Navigate to the path where the Information Center is installed. By default, the *DB2 Information Center* is installed in the /opt/ibm/db2ic/V9.7 directory.

b. Navigate from the installation directory to the doc/bin directory.

c. Run the help_start script:

```
help_start
```

The systems default Web browser opens to display the stand-alone Information Center.

3. Click the **Update** button ( ). (JavaScript must be enabled in your browser.) On the right panel of the Information Center, click **Find Updates**. A list of updates for existing documentation displays.
4. To initiate the installation process, check the selections you want to install, then click **Install Updates**.
5. After the installation process has completed, click **Finish**.
6. Stop the stand-alone Information Center:
   - On Windows, navigate to the installation directory's doc\bin directory, and run the help_end.bat file:

     ```
     help_end.bat
     ```

     Note: The help_end batch file contains the commands required to safely stop the processes that were started with the help_start batch file. Do not use Ctrl-C or any other method to stop help_start.bat.
   - On Linux, navigate to the installation directory's doc/bin directory, and run the help_end script:

     ```
     help_end
     ```

     Note: The help_end script contains the commands required to safely stop the processes that were started with the help_start script. Do not use any other method to stop the help_start script.
7. Restart the *DB2 Information Center*.
   - On Windows, click **Start** > **Control Panel** > **Administrative Tools** > **Services**. Then right-click **DB2 Information Center** service and select **Start**.
   - On Linux, enter the following command:

     ```
     /etc/init.d/db2icdv97 start
     ```

Results

The updated *DB2 Information Center* displays the new and updated topics.

DB2 tutorials

The DB2 tutorials help you learn about various aspects of DB2 products. Lessons provide step-by-step instructions.

Before you begin

You can view the XHTML version of the tutorial from the Information Center at http://publib.boulder.ibm.com/infocenter/db2help/.

Some lessons use sample data or code. See the tutorial for a description of any prerequisites for its specific tasks.

DB2 tutorials

To view the tutorial, click the title.

"pureXML®" in *pureXML Guide*
Set up a DB2 database to store XML data and to perform basic operations with the native XML data store.

"Visual Explain" in *Visual Explain Tutorial*
Analyze, optimize, and tune SQL statements for better performance using Visual Explain.

DB2 troubleshooting information

A wide variety of troubleshooting and problem determination information is available to assist you with using DB2 database products.

DB2 documentation
Troubleshooting information can be found in the *Troubleshooting and Tuning Database Performance* or the Database fundamentals section of the *DB2 Information Center*. The troubleshooting information contains topics that can help you isolate and identify problems with DB2 diagnostic tools and utilities. There are also solutions to some of the most common problems and advice on how to solve problems you might encounter with your DB2 database products.

IBM Support Portal
See the IBM Support Portal if you are experiencing problems and want help finding possible causes and solutions. The Technical Support site has links to the latest DB2 publications, TechNotes, Authorized Program Analysis Reports (APARs or bug fixes), fix packs, and other resources. You can search through this knowledge base to find possible solutions to your problems.

Access the IBM Support Portal at http://www.ibm.com/support/entry/portal/Overview/Software/Information_Management/DB2_for_Linux,_UNIX_and_Windows.

Terms and Conditions

Permissions for the use of these publications is granted subject to the following terms and conditions.

Personal use: You may reproduce these Publications for your personal, non commercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these Publications, or any portion thereof, without the express consent of IBM.

Commercial use: You may reproduce, distribute and display these Publications solely within your enterprise provided that all proprietary notices are preserved.

You may not make derivative works of these Publications, or reproduce, distribute or display these Publications or any portion thereof outside your enterprise, without the express consent of IBM.

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the Publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the Publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Appendix B. Notices

This information was developed for products and services offered in the U.S.A. Information about non-IBM products is based on information available at the time of first publication of this document and is subject to change.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information about the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country/region where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions; therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements, changes, or both in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those

© Copyright IBM Corp. 1993, 2012

websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information that has been exchanged, should contact:

IBM Canada Limited
U59/3600
3600 Steeles Avenue East
Markham, Ontario L3R 9Z7
CANADA

Such information may be available, subject to appropriate terms and conditions, including, in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems, and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements, or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information may contain examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious, and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating

platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work must include a copyright notice as follows:

© (*your company name*) (*year*). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. *_enter the year or years_*. All rights reserved.

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml.

The following terms are trademarks or registered trademarks of other companies

- Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
- Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle, its affiliates, or both.
- UNIX is a registered trademark of The Open Group in the United States and other countries.
- Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.
- Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Index

Special characters

_DETAILS table functions 9
.db2top configuration file 280
.db2toprc configuration file 280

A

activities
  monitor elements
    act_aborted_total 494
    act_completed_total 495
    act_rejected_total 497
    act_total 501
    activity_collected 502
    activity_id 502
    activity_secondary_id 503
    activity_state 504
    activity_type 504
    coord_act_aborted_total 584
    coord_act_completed_total 584
    coord_act_rejected_total 589
    parent_activity_id 797
activity event monitor
  monitor data returned in XML documents 9
Activity Monitor
  overview 367
  setup 373
ACTIVITYTOTALTIME activity threshold
  monitor elements
    activitytotaltime_threshold_id 505
    activitytotaltime_threshold_value 505
    activitytotaltime_threshold_violated 506
administrative views
  APPL_PERFORMANCE
    scenario 370
  BP_HITRATIO
    scenario 372
  BP_READ_IO
    scenario 372
  BP_WRITE_IO
    scenario 372
  LONG_RUNNING_SQL
    scenario 370
  QUERY_PREP_COST
    scenario 370
  TOP_DYNAMIC_SQL
    scenario 370
agents
  monitor elements
    agent_id 507
    agent_id_holding_lock 508
    agent_pid 509
    agent_status 509
    agent_sys_cpu_time 509
    agent_usr_cpu_time 510
    agent_wait_time 511
    agent_waits_total 512
    agents_created_empty_pool 513
    agents_from_pool 513
    agents_registered 514
agents *(continued)*
  monitor elements *(continued)*
    agents_registered_top 514
    agents_stolen 514
    agents_top 515
    agents_waiting_on_token 515
    agents_waiting_top 516
    appl_priority 523
    associated_agents_top 530
    coord_agent_pid 589
    coord_agents_top 589
    idle_agents 692
    locks_waiting 740
    max_agent_overflows 748
    num_agents 771
    num_assoc_agents 771
    priv_workspace_size_top 864
    quiescer_agent_id 871
    rolled_back_agent_id 887
alert actions
  health indicators
    states 332
alert thresholds
  configuring 330
alerts
  enabling 303
  resolving
    GET RECOMMENDATIONS command 322
    Health Center 323
    SQL queries 318
  retrieving recommendations
    client application 318
aliases
  input_db_alias element monitor element 695
analyzedata
  monitoring 230
API request types
  health monitor 362
  snapshot monitor 270
applications
  monitor elements
    appl_id 519
    appl_id_holding_lk 521
    appl_id_oldest_xact 522
    appl_idle_time 522
    appl_name 523
    appl_priority 523
    appl_priority_type 524
    appl_section_inserts 525
    appl_section_lookups 525
    appl_status 525
    application_handle 528
    appls_cur_cons 529
    appls_in_db2 529
    client_applname 555
    creator 601
    rolled_back_participant_no 887
    tpmon_client_app 1040
attributes
  progress_list_attr monitor element 866

© Copyright IBM Corp. 1993, 2012

audits
monitor elements
audit_events_total 531
audit_file_write_wait_time 532
audit_file_writes_total 533
authorization IDs
monitor elements
auth_id 537
execution_id 639
quiescer_auth_id 871
session_auth_id 910
authorization level monitor element 538
automatic storage paths
monitor elements
db_storage_path 610
sto_path_free_sz 946

B

backups
databases
Database Backup Required health indicator 350
db.db_backup_req health indicator 350
last_backup monitor element 711
books
ordering 1076
buffer pools
monitor elements
activity 295
automatic 540
block_ios 541
bp_cur_buffsz 543
bp_id 543
bp_name 543
bp_new_buffsz 544
bp_pages_left_to_remove 544
bp_tbsp_use_count 544
buff_free 545
buff_free_bottom 545
pool_async_data_read_reqs 806
pool_async_data_reads 807
pool_async_data_writes 808
pool_async_index_read_reqs 808
pool_async_index_reads 809
pool_async_index_writes 810
pool_async_read_time 811
pool_async_write_time 812
pool_async_xda_read_reqs 812
pool_async_xda_reads 813
pool_async_xda_writes 814
pool_data_l_reads 816
pool_data_p_reads 818
pool_data_writes 820
pool_drty_pg_steal_clns 822
pool_drty_pg_thrsh_clns 823
pool_index_l_reads 825
pool_index_p_reads 827
pool_index_writes 829
pool_lsn_gap_clns 831
pool_no_victim_buffer 832
pool_read_time 833
pool_temp_data_l_reads 835
pool_temp_data_p_reads 837
pool_temp_index_l_reads 839
pool_temp_index_p_reads 841
pool_temp_xda_l_reads 843
pool_temp_xda_p_reads 845
buffer pools *(continued)*
monitor elements *(continued)*
pool_write_time 847
pool_xda_l_reads 849
pool_xda_p_reads 851
pool_xda_writes 853
tablespace_cur_pool_id 956
tablespace_next_pool_id 962
tbsp_cur_pool_id 956
tbsp_next_pool_id 962
monitoring
administrative views 372
BUFFERPOOLS event type
details 23
buffers
num_log_data_found_in_buffer monitor elements 777
byte order
byte_order monitor element 547

C

caching
stats_cache_size monitor element 928
catalog cache
db.catcache_hitratio health indicator 356
monitor elements
cat_cache_inserts 547
cat_cache_lookups 548
cat_cache_overflows 550
cat_cache_size_top 551
catalog nodes
monitor elements
catalog_node 551
catalog_node_name 552
client applications
health snapshots 310
client operating platform monitor element 559
client process ID monitor element 558
client product and version ID monitor element 560
code pages
monitor elements
codepage_id 564
host_ccsid 681
coded character set identifier (CCSID)
host_ccsid monitor element 681
collecttext
statement 234
command line processor (CLP)
commands
health monitor 362
health snapshot capturing 309
commit statements attempted monitor element 565
commits
int_commits monitor element 698
communication error time monitor element
gw_comm_error_time element 666
communication errors monitor element
gw_comm_errors element 666
communication protocols
client_protocol monitor element 561
completed progress work units monitor element 865
component elapsed times
monitor elements
overview 241, 413
viewing
activity-level examples 252
system-level examples 248

component processing times
monitor elements
overview 241, 413
viewing
activity-level examples 252
system-level examples 248
con_response_time monitor element 567
configuration
.db2toprc file 280
connections
monitor elements
appl_con_time 519
appls_cur_cons 529
appls_in_db2 529
con_elapsed_time 567
con_local_dbases 567
conn_complete_time 578
conn_time 578
connection_status 579
connections_top 580
dl_conns 632
gw_connections_top 667
gw_cons_wait_client 667
gw_cons_wait_host 667
gw_cur_cons 668
gw_total_cons 669
local_cons 715
local_cons_in_exec 716
num_gw_conn_switches 775
rem_cons_in 875
rem_cons_in_exec 876
total_sec_cons 1027
CONNECTIONS event type
overview 23
containers
monitor elements
container_accessible 581
container_id 581
container_name 581
container_total_pages 582
container_type 583
container_usable_pages 583
control tables 204
event monitors 204
counters
data element type 290
CPU time
monitor elements
agent_sys_cpu_time 509
agent_usr_cpu_time 510
ss_sys_cpu_time 926
ss_usr_cpu_time 926
stmt_sys_cpu_time 940
stmt_usr_cpu_time 943
system_cpu_time 950
total_cpu_time 1001
total_sys_cpu_time 1037
total_usr_cpu_time 1038
user_cpu_time 1053
CPU utilization
monitor elements
cpu_idle 593
cpu_iowait 594
cpu_system 596
cpu_usage_total 597
cpu_user 598
CREATE EVENT MONITOR statement
event types 23
creator monitor element 601
cursors
monitor elements
acc_curs_blk 494
blocking_cursor 542
cursor_name 603
open_cursors 782
open_loc_curs 783
open_loc_curs_blk 783
open_rem_curs 784
open_rem_curs_blk 784
rej_curs_blk 875

D

data
element types
counters 290
overview 289
data objects
monitoring 411
data partitions
data_partition_id monitor element 604
data sources
data source name monitor element 605
health indicator 359
DATABASE event type 23
database paths
db_path element monitor element 609
database system monitor
data organization 289
information restricting 283
interfaces 296
memory requirements 292
output 291
sample 296
self-describing data stream 291
database-managed space (DMS)
table spaces
health indicators 338
databases
aliases
application monitor element 556
gateway monitor element 668
connections
connects since database activation monitor element 1000
local
con_local_dbases monitor element 567
monitor elements
application 556
connects since database activation 1000
database deactivation timestamp 631
gateway 668
monitoring
overview 3
datasource_name element 605
db_heap_top monitor element 607
db.lock_escal_rate health indicator 355
db.locklist_utilization health indicator 354
DB2 Connect
monitor elements
gw_con_time 666
gw_cur_cons 668
gw_exec_time 668

DB2 Connect *(continued)*
monitor elements *(continued)*
gw_total_cons 669
DB2 Information Center
languages 1078
updating 1079, 1080
versions 1077
DB2 Performance Counters 386
DB2 workload manager
monitor elements
total queue assignments 1059
total queue time 1060
db2advis command
input file
creating with the package cache event monitor 76
db2event.ctl control file 210
db2evmonfmt tool 71
details 30
lock event data 37
unit of work event data 96
db2perfc command
resetting database performance values 388
db2perfi command
installing and registering DB2Perf.DLL 386
db2perfr command
registering administrator user name and password with DB2 386
db2top command
monitoring 278
deadlocks
db.deadlock_rate health indicator 353
event types 23
monitor elements
deadlock_id 613
deadlock_node 614
deadlocks 614
dl_conns 632
int_deadlock_rollbacks 699
participant_no 798
reports 37
DELETE statement
delete_sql_stmts monitor element 617
deprecated functionality
health indicators 299, 333
health monitor 307
health monitor CLP commands 362
health monitor interfaces 359
health monitor SQL table functions 361
descriptors
progress_description monitor element 865
Design Advisor
creating input file with the package cache event monitor 76
DETAILS.XML
monitor table functions 9
determine
category 227
phase 227
problem 227
diagnose
slowdowns 227
SQL performance 227
disconn_time element 631
documentation
overview 1073
PDF files 1073
printed 1073
documentation *(continued)*
terms and conditions of use 1082

E

environment handles
comp_env_desc monitor element 565
errors
gw_comm_errors monitor element 666
event monitor
package listing
unit of work event monitor 103
unit of work 94
event monitors
blocked 212
buffers 212
control tables 204
creating
file 208
overview 200
partitioned database 214
pipe 213
table 201
data transfers between systems 220
database system events 198
db2evmonfmt Java-based tool for parsing data 30
DEADLOCK WITH DETAILS HISTORY 382
elements
count 592
event_monitor_name 637
evmon_activates 638
evmon_flushes 638
event type to logical data group mappings 454
file management 210
locking
overview 35
usage example 37
logical data groups
target tables 204
named pipe management 214
non-blocked 212
overflow records 204
output
sample 217
self-describing data stream 218
overflow records 204
overview 197
records 217
statistics
data collected in XML format 128
logical data groups 128
monitor elements 128
table management 204
unformatted event table 30
unit of work
usage example 98
events
information collection 198
monitor elements
event_time 637
start_time 927
stop_time 947
records 217

F

fast communication manager (FCM)
  monitor elements
    buff_auto_tuning 544
    buff_free 545
    buff_free_bottom 545
    buff_max 546
    buff_total 546
    ch_auto_tuning 552
    ch_free 552
    ch_free_bottom 553
    ch_max 553
    ch_total 554
    fcm_message_recv_volume 640
    fcm_message_recv_wait_time 642
    hostname 683
    remote_member 877
    total_buffers_rcvd 994
    total_buffers_sent 994
FCM (Fast Communications Manager)
  monitoring 8
  wait time monitor elements 246
federated server monitor elements
  disconnects 631
fetching
  fetch_count monitor element 661
file event monitors
  buffering 212
  creating 208
  formatting output from command line 217
  managing 210
file systems
  db.log_fs_util health indicator 352
  monitor elements
    fs_caching 664
    fs_id 664
    fs_total_size 665
    fs_used_size 665
files
  files_closed monitor element 662
FLUSH EVENT MONITOR statement
  event types 23

G

GET SNAPSHOT command
  sample output 272, 373
global health snapshots 315
global snapshots on partitioned database systems 275
gw_db_alias element 668

H

hash joins
  monitor elements
    active_hash_joins 501
    hash_join_overflows 678
    hash_join_small_overflows 679
    post_shrthreshold_hash_joins 855
    post_threshold_hash_joins 857
    total_hash_joins 1002
health alerts
  enabling 303
  recommendations 318
  resolving
    client applications 322
health alerts *(continued)*
  resolving *(continued)*
    SQL queries 318
Health Center
  health indicators 299, 333
  overview 305, 315
  status beacon 315
  tasks 305
health indicators
  alert actions 332
  alerts
    resolving using SQL 318
    resolving using the Health Center 323
    retrieving recommendations 318, 322
  applications waiting on locks 355
  catalog cache hit ratio 356
  collection state-based 299, 333
  configuring
    client applications 328
    Health Center 330
    overview 324
    resetting 328
    retrieving 326
    updates 327
  data 308
  databases
    heap utilization 358
    highest severity alert state 349
    operational state 348
  db.alert_state 349
  db.apps_waiting_locks 355
  db.catcache_hitratio 356
  db.db_auto_storage_util 339
  db.db_backup_req 350
  db.db_heap_util 358
  db.db_op_status 348
  db.deadlock_rate 353
  db.fed_nicknames_op_status 358
  db.fed_servers_op_status 359
  db.hadr_delay 351
  db.hadr_op_status 351
  db.lock_escal_rate 355
  db.locklist_utilization 354
  db.log_fs_util 352
  db.log_util 352
  db.max_sort_shrmem_util 346
  db.pkgcache_hitratio 356
  db.shrworkspace_hitratio 357
  db.sort_shrmem_util 345
  db.spilled_sorts 346
  db.tb_reorg_req 349
  db.tb_runstats_req 350
  db2.db2_alert_state 348
  db2.db2_op_status 347
  db2.mon_heap_util 357
  db2.sort_privmem_util 344
  deadlock rate 353
  DMS table spaces 338
  format 338
  instances
    highest severity alert state 348
    operational state 347
  lock escalation rate 355
  lock list utilization 354
  logs
    file system utilization 352
    space utilization 352

health indicators *(continued)*
monitor heap utilization 357
overview 299, 333
package cache hit ratio 356
process cycle 302
shared workspace hit ratio 357
sort memory utilization
long-term shared 346
private 344
shared 345
sorts that overflowed 346
state-based 299, 333
summary 336
table spaces
container operational state 344
container utilization 342
operational state 343
storage utilization 341
threshold-based 299, 333
ts.ts_auto_resize_status 340
ts.ts_op_status 343
ts.ts_util 341
ts.ts_util_auto_resize 341
tsc.tscont_op_status 344
tsc.utilization 342
health monitor
alerts 324
API request types 362
CLP commands 362
details 299
graphical tools 315
Health Center 315
Health Center Status Beacon 315
interfaces 359
logical data groups 334
recommendation retrieval
using client application 322
using CLP 318
using SQL 318
sample output 313
SQL table functions 361
starting 307
stopping 307
thresholds 324
health snapshots
capturing
using client applications 310
using CLP 309
using SQL table functions 309
global 315
help
configuring language 1078
SQL statements 1077
hierarchy
time-spent monitor elements 241, 413
High Availability Disaster Recovery (HADR)
health indicators
db.hadr_delay 351
db.hadr_op_status 351
monitor elements
hadr_connect_status 669
hadr_connect_time 670
hadr_heartbeat 670
hadr_local_host 671
hadr_local_service 671
hadr_log_gap 672
hadr_peer_window 672
High Availability Disaster Recovery (HADR) *(continued)*
monitor elements *(continued)*
hadr_peer_window_end 672
hadr_primary_log_file 673
hadr_primary_log_lsn 673
hadr_primary_log_page 674
hadr_remote_host 674
hadr_remote_instance 674
hadr_remote_service 675
hadr_role 675
hadr_standby_log_file 676
hadr_standby_log_lsn 676
hadr_standby_log_page 676
hadr_state 677
hadr_syncmode 677
hadr_timeout 678
histograms
monitor elements
histogram_type 679
number_in_bin 782
top 987
host databases
host_db_name monitor element 681
name monitor element 681

I

I/O
monitor elements
num_log_part_page_io 777
num_log_read_io 778
num_log_write_io 778
num_pages_from_block_IOs 795
num_pages_from_vectored_IOs 796
vectored_ios 1057
identifiers
monitor elements
arm_correlator 530
bin_id 540
db_work_action_set_id 611
db_work_class_id 611
host_prdid 681
sc_work_action_set_id 898
sc_work_class_id 899
service_class_id 907
sql_req_id 919
work_action_set_id 1062
work_class_id 1063
identify
scope 230
SQL statement 233
identify slow down problem 232
indexes
index object pages monitor element 693
monitor elements
iid 692
index_object_pages 693
index_only_scans 694
index_scans 694
index_tbsp_id 694
int_node_splits 700
nleaf 770
nlevels 770
page_allocations 794
pages_merged 796
reorg_index_id monitor 878
root_node_splits 888

Indoubt Transaction Manager
overview 389
indoubt transactions
monitoring 389
insert_timestamp monitor element 696
inserting data
appl_section_inserts monitor element 525
instances
operational state health indicator 347
int_rows_deleted monitor element 702
isolation levels
effective_isolation monitor element 633

J

java tools
db2evmonfmt 71

L

large objects (LOBs)
lob_object_pages element 715
list
top ranked 232
location monitor element 607
lock escalation
db.lock_escal_rate health indicator 355
lock_escalation monitor element 720
lock list utilization health indicator 354
lock modes
monitor elements
lock_current_mode 719
lock_mode 724
lock_mode_requested 725
lock_escalation monitor element 720
locks
monitor elements
agent_id_holding_lock 508
appl_id_holding_lk 521
effective_lock_timeout 634
lock_attributes 717
lock_count 718
lock_escalation 720
lock_escals 721
lock_hold_count 723
lock_list_in_use 723
lock_name 726
lock_node 727
lock_object_name 727
lock_object_type 728
lock_release_flags 730
lock_status 730
lock_timeout_val 731
lock_timeouts 732
lock_wait_time 734
lock_waits 737
locks_held 738
locks_held_top 739
locks_in_list 740
locks_waiting 740
participant_no_holding_lk 799
remote_lock_time 876
remote_locks 877
sequence_no_holding_lk 904
stmt_lock_timeout 934
uow_lock_wait_time 1049
locks *(continued)*
monitor elements *(continued)*
x_lock_escals 1066
monitoring 8
timeouts
reports 37
waits
lock_wait_end_time monitor element 733
lock_wait_start_time monitor element 734
reports 37
log buffers
num_log_buffer_full monitor element 775
log disk monitor elements
log_disk_wait_time 741
log_disk_waits_total 743
log sequence numbers (LSNs)
monitor elements
hadr_primary_log_lsn 673
hadr_standby_log_lsn 676
logical data groups
COLLECT ACTIVITY DATA settings effects 491
data organization 289
event monitors 457
health monitor 334
mapping to event types 454
snapshot monitor 419
target tables 204
logs
health indicators
db.log_fs_util 352
db.log_util 352
monitor elements
current_active_log 602
current_archive_log 602
diaglog_write_wait_time 618
diaglog_writes_total 619
first_active_log 663
hadr_log_gap 672
hadr_primary_log_file 673
hadr_primary_log_page 674
hadr_standby_log_file 676
hadr_standby_log_page 676
last_active_log 711
log_held_by_dirty_pages 744
log_read_time 744
log_reads 745
log_to_redo_for_recovery 745
log_write_time 746
log_writes 746
sec_log_used_top 899
sec_logs_allocated 900
smallest_log_avail_node 913
tot_log_used_top 987
total_log_available 1009
total_log_used 1010
uow_log_space_used 1049
long data
long_object_pages monitor element 747

M

memory
health indicators
db.sort_shrmem_util 345
db2.sort_privmem_util 344
monitor elements
comm_private_mem 564

memory *(continued)*
monitor elements *(continued)*
db_heap_top 607
lock_list_in_use 723
pool_config_size 815
pool_cur_size 815
pool_id 824
pool_secondary_id 834
pool_watermark 847
monitoring
overview 8
requirements
database system monitor 292
Memory Visualizer
overview 364
using 362
messages
monitor elements
message 767
message_time 768
metrics
See also monitor elements
data objects 411
ranking monitor elements returned in XML documents 18
returned by event monitors 9
MON_FORMAT_ table functions
comparison with XMLTABLE table function 14
viewing monitor elements as rows in a table 18
mon_heap_sz database manager configuration parameter
overview 292
monitor element
routines
total_routine_user_code_proc_time 1020
monitor elements
acc_curs_blk 494
act_exec_time 497
act_remapped_in
details 498
act_remapped_out
details 499
act_rqsts_total 500
activation time
last_wlm_reset 714
active_sorts 502
activities
act_aborted_total 494
act_completed_total 495
act_rejected_total 497
act_total 501
activity_collected 502
activity_id 502
activity_secondary_id 503
activity_state 504
activity_type 504
coord_act_aborted_total 584
coord_act_completed_total 584
coord_act_rejected_total 589
overview 409
parent_activity_id 797
activity_metrics 129, 163
ACTIVITYTOTALTIME activity threshold
activitytotaltime_threshold_id 505
activitytotaltime_threshold_value 505
activitytotaltime_threshold_violated 506
adapter_name 506
address 506
monitor elements *(continued)*
agents
agent_id 507
agent_id_holding_lock 508
agent_pid 509
agent_status 509
agent_sys_cpu_time 509
agent_usr_cpu_time 510
agent_wait_time 511
agent_waits_total 512
agents_created_empty_pool 513
agents_from_pool 513
agents_registered 514
agents_registered_top 514
agents_stolen 514
agents_top 515
agents_waiting_on_token 515
agents_waiting_top 516
appl_priority 523
associated_agents_top 530
coord_agent_pid 589
coord_agents_top 589
idle_agents 692
max_agent_overflows 748
num_agents 771
num_assoc_agents 771
priv_workspace_size_top 864
quiescer_agent_id 871
rolled_back_agent_id 887
agg_temp_tablespace_top 516
aliases
client_db_alias 556
input_db_alias 695
applications
appl_id 519
appl_id_holding_lk 521
appl_id_oldest_xact 522
appl_idle_time 522
appl_name 523
appl_priority_type 524
appl_section_inserts 525
appl_section_lookups 525
appl_status 525
application_handle 528
client_applname 555
memory_pool_used 764
tpmon_client_app 1040
attributes
progress_list_attr 866
audits
audit_events_total 531
audit_file_write_wait_time 532
audit_file_writes_total 533
auth_id 537
authority_bitmap 537
authorization IDs
execution_id 639
session_auth_id 910
automatic storage path
sto_path_free_sz 946
binds_precompiles 540
blocking_cursor 542
blocks_pending_cleanup 542
boundary_leaf_node_splits 543
buffer pools
activity 295
automatic 540

monitor elements *(continued)*
buffer pools *(continued)*
block_ios 541
bp_cur_buffsz 543
bp_id 543
bp_name 543
bp_new_buffsz 544
bp_pages_left_to_remove 544
bp_tbsp_use_count 544
buff_free 545
buff_free_bottom 545
pool_async_data_read_reqs 806
pool_async_data_reads 807
pool_async_data_writes 808
pool_async_index_read_reqs 808
pool_async_index_reads 809
pool_async_index_writes 810
pool_async_read_time 811
pool_async_write_time 812
pool_async_xda_read_reqs 812
pool_async_xda_reads 813
pool_async_xda_writes 814
pool_data_l_reads 816
pool_data_p_reads 818
pool_data_writes 820
pool_drty_pg_steal_clns 822
pool_drty_pg_thrsh_clns 823
pool_index_l_reads 825
pool_index_p_reads 827
pool_index_writes 829
pool_lsn_gap_clns 831
pool_no_victim_buffer 832
pool_read_time 833
pool_temp_data_l_reads 835
pool_temp_data_p_reads 837
pool_temp_index_l_reads 839
pool_temp_index_p_reads 841
pool_temp_xda_l_reads 843
pool_temp_xda_p_reads 845
pool_write_time 847
pool_xda_l_reads 849
pool_xda_p_reads 851
pool_xda_writes 853
buffers
num_log_data_found_in_buffer 777
byte order
byte_order 547
caches
stats_cache_size 928
cat_cache_inserts 547
cat_cache_lookups 548
cat_cache_overflows 550
cat_cache_size_top 551
catalog_node 551
catalog_node_name 552
client_hostname 557
client_pid 558
client_platform 559
client_port_number 560
client_prdid 560
code pages
codepage_id 564
host_ccsid 681
comm_private_mem 564
commit_sql_stmts 565
commits
int_commits 698
monitor elements *(continued)*
communication protocols
client_protocol 561
component elapsed time spent 241, 413
component processing time spent 241, 413
CONCURRENTDBCOORDACTIVITIES threshold
concurrentdbcoordactivities_wl_was _threshold_id 574
concurrentdbcoordactivities_wl_was _threshold_queued 575
concurrentdbcoordactivities_wl_was _threshold_value 575
concurrentdbcoordactivities_wl_was _threshold_violated 576
connection_start_time 579
connections
appl_con_time 519
appls_cur_cons 529
appls_in_db2 529
con_elapsed_time 567
con_local_dbases 567
conn_complete_time 578
conn_time 578
connection_status 579
connections_top 580
gw_connections_top 667
gw_cons_wait_client 667
gw_cons_wait_host 667
gw_cur_cons 668
gw_total_cons 669
local_cons 715
local_cons_in_exec 716
num_gw_conn_switches 775
rem_cons_in 875
rem_cons_in_exec 876
total_cons 1000
total_sec_cons 1027
contained in XML documents
formatting 18
containers
container_accessible 581
container_id 581
container_name 581
container_total_pages 582
container_type 583
container_usable_pages 583
coord_act_est_cost_avg 584
coord_act_exec_time_avg 585
coord_act_interarrival_time_avg 586
coord_act_lifetime_avg 587
coord_act_queue_time_avg 588
coord_member 590
country_code
see monitor elements, territory_code 981
CPU time
ss_sys_cpu_time 926
ss_usr_cpu_time 926
stmt_sys_cpu_time 940
stmt_usr_cpu_time 943
system_cpu_time 950
total_cpu_time 1001
total_sys_cpu_time 1037
total_usr_cpu_time 1038
user_cpu_time 1053
cpu_configured 593
cpu_cores_per_socket 593
cpu_hmt_degree 593
cpu_idle 593

monitor elements *(continued)*
cpu_iowait 594
cpu_load_long 595
cpu_load_medium 595
cpu_load_short 595
cpu_online 596
cpu_speed 596
cpu_system 596
cpu_timebase 597
cpu_total 597
cpu_usage_total 597
cpu_user 598
cursors
cursor_name 603
rej_curs_blk 875
data organization 289
data_object_l_pages - Table data logical pages 604
database manager
server_db2_type 905
database paths
db_path 609
database system 493
db_heap_top 607
db_storage_path 610
DB2 Connect
gw_con_time 666
gw_exec_time 668
db2_process_identifier 605
db2_process_name 606
deadlocks
deadlock_id 613
deadlock_node 614
deadlocks 614
dl_conns 632
int_deadlock_rollbacks 699
del_keys_cleaned 617
DELETE statement
delete_sql_stmts 617
descriptors
progress_description 865
destination_service_class_id 618
edu_id 633
eff_stmt_text 633
empty_pages_deleted 635
empty_pages_reused 635
environment handles
comp_env_desc 565
errors
gw_comm_errors 666
event monitors
count 592
event_monitor_name 637
evmon_activates 638
evmon_flushes 638
list 457
events
event_time 637
start_time 927
stop_time 947
executable_id 639
fabrications
stats_fabricate_time 929
stats_fabrications 929
fast communication manager (FCM
buff_auto_tuning 544
ch_auto_tuning 552

monitor elements *(continued)*
fast communication manager (FCM)
buff_max 546
buff_total 546
ch_free 552
ch_free_bottom 553
ch_max 553
ch_total 554
fcm_message_recv_volume 640
fcm_message_recv_wait_time 642
hostname 683
remote_member 877
total_buffers_rcvd 994
total_buffers_sent 994
federated servers
disconnects 631
fetching
fetch_count 661
file systems
fs_caching 664
fs_id 664
fs_total_size 665
fs_used_size 665
files
files_closed 662
gw_comm_error_time 666
hash joins
active_hash_joins 501
hash_join_overflows 678
hash_join_small_overflows 679
post_shrthreshold_hash_joins 855
post_threshold_hash_joins 857
total_hash_joins 1002
high availability disaster recovery (HADR)
hadr_connect_status 669
hadr_connect_time 670
hadr_heartbeat 670
hadr_local_host 671
hadr_local_service 671
hadr_log_gap 672
hadr_peer_window 672
hadr_peer_window_end 672
hadr_primary_log_file 673
hadr_primary_log_lsn 673
hadr_primary_log_page 674
hadr_remote_host 674
hadr_remote_instance 674
hadr_remote_service 675
hadr_role 675
hadr_standby_log_file 676
hadr_standby_log_lsn 676
hadr_standby_log_page 676
hadr_state 677
hadr_syncmode 677
hadr_timeout 678
histograms
histogram_type 679
number_in_bin 782
top 987
host databases
host_db_name 681
host_name 683
I/O
num_log_part_page_io 777
num_log_read_io 778
num_log_write_io 778
num_pages_from_block_IOs 795

monitor elements *(continued)*
I/O *(continued)*
num_pages_from_vectored_IOs 796
vectored_ios 1057
ida_recv_volume 683
ida_recv_wait_time 685
ida_recvs_total 686
ida_send_volume 687
ida_send_wait_time 689
ida_sends_total 690
identifiers
arm_correlator 530
bin_id 540
db_work_action_set_id 611
db_work_class_id 611
host_prdid 681
sc_work_action_set_id 898
sc_work_class_id 899
service_class_id 907
sql_req_id 919
work_action_set_id 1062
work_class_id 1063
inbound_bytes_received 692
inbound_bytes_sent 692
inbound_comm_address 693
include_col_updates 693
index_object_l_pages - Index data logical pages 694
indexes
iid 692
index_object_pages 693
index_only_scans 694
index_scans 694
index_tbsp_id 694
int_node_splits 700
nleaf 770
nlevels 770
page_allocations 794
pages_merged 796
root_node_splits 888
insert_timestamp 696
int_rows_deleted 702
interfaces for viewing metrics in XML documents 14
interfaces that return XML documents 9
is_system_appl 710
isolation levels
effective_isolation 633
key_updates 710
large objects (LOBs)
lob_object_pages 715
last_executable_id 712
last_request_type 713
lob_object_l_pages - LOB data logical pages 715
locations
db_location 607
lock modes
lock_current_mode 719
lock_mode 724
lock_mode_requested 725
locking
hld_application_handle 680
hld_member 681
req_agent_tid 883
req_application_handle 883
req_executable_id 883
req_member 884
locks
effective_lock_timeout 634
monitor elements *(continued)*
locks *(continued)*
lock_attributes 717
lock_count 718
lock_escals 721
lock_hold_count 723
lock_list_in_use 723
lock_name 726
lock_node 727
lock_object_name 727
lock_object_type 728
lock_release_flags 730
lock_status 730
lock_timeout_val 731
lock_timeouts 732
lock_wait_time 734
lock_waits 737
locks_held 738
locks_held_top 739
locks_in_list 740
locks_waiting 740
participant_no_holding_lk 799
remote_lock_time 876
remote_locks 877
sequence_no_holding_lk 904
stmt_lock_timeout 934
uow_lock_wait_time 1049
written from EVMON_FORMAT_UE_TO_TABLES table function 64
written from EVMON_FORMAT_UE_TO_XML table function 40
x_lock_escals 1066
log buffers
num_log_buffer_full 775
log files
current_active_log 602
current_archive_log 602
diaglog_write_wait_time 618
diaglog_writes_total 619
first_active_log 663
last_active_log 711
log_read_time 744
log_reads 745
sec_logs_allocated 900
log space
log_held_by_dirty_pages 744
log_to_redo_for_recovery 745
log_write_time 746
log_writes 746
sec_log_used_top 899
smallest_log_avail_node 913
tot_log_used_top 987
total_log_available 1009
total_log_used 1010
uow_log_space_used 1049
log writing to disk
log_disk_wait_time 741
log_disk_waits_total 743
logical data groups 423
long data
long_object_pages 747
long_object_l_pages - Long object data logical pages 747
machine_identification 748
member 760
memory_free 762
memory_pool_id 762
memory_pool_type 762

monitor elements *(continued)*
memory_pool_used_hwm 764
memory_set_committed 765
memory_set_id 765
memory_set_size 765
memory_set_type 765
memory_set_used 766
memory_set_used_hwm 766
memory_swap_free 766
memory_swap_total 767
memory_total 767
messages
message 767
metrics
ranking 18
names
db_name 608
dcs_db_name 612
service_subclass_name 908
service_superclass_name 909
work_action_set_name 1062
work_class_name 1063
network time
max_network_time_1_ms 759
max_network_time_100_ms 758
max_network_time_16_ms 758
max_network_time_4_ms 759
max_network_time_500_ms 760
max_network_time_gt500_ms 760
network_time_bottom 769
network_time_top 769
network_time_bottom 769
network_time_top 769
nicknames
create_nickname 600
create_nickname_time 601
nodes
coord_node 590
node_number 770
num_nodes_in_db2_instance 780
ss_node_number 925
nonboundary_leaf_node_splits 771
num_db_storage_paths 773
num_exec_with_metrics 773
num_indoubt_trans 775
num_nodes_in_db2_instance 780
num_remaps 780
num_transmissions 781
num_transmissions_group 781
numbers
progress_list_cur_seq_num 866
ss_number 925
OLAP
active_olap_funcs 501
olap_func_overflows 782
post_threshold_olap_funcs 858
total_olap_funcs 1010
open_cursors 782
open_loc_curs 783
open_loc_curs_blk 783
open_rem_curs 784
open_rem_curs_blk 784
operations
direct_read_reqs 620
direct_read_time 622
direct_reads 624
direct_write_reqs 626
monitor elements *(continued)*
operations *(continued)*
direct_write_time 627
direct_writes 629
stmt_operation 935
os_level 785
os_name 785
os_release 785
os_version 785
outbound bytes
max_data_sent_1024 753
max_data_sent_128 753
max_data_sent_16384 754
max_data_sent_2048 754
max_data_sent_256 755
max_data_sent_31999 755
max_data_sent_4096 756
max_data_sent_512 756
max_data_sent_64000 757
max_data_sent_8192 757
max_data_sent_gt64000 757
outbound bytes received
max_data_received_1024 748
max_data_received_128 749
max_data_received_16384 749
max_data_received_2048 750
max_data_received_256 750
max_data_received_31999 751
max_data_received_4096 751
max_data_received_512 751
max_data_received_64000 752
max_data_received_8192 752
max_data_received_gt64000 753
outbound_bytes_received 786
outbound_bytes_received_bottom 787
outbound_bytes_received_top 787
outbound bytes sent
outbound_bytes_sent 788
outbound_bytes_sent_bottom 788
outbound_bytes_sent_top 788
outbound communication
outbound_appl_id 786
outbound_comm_address 788
outbound_comm_protocol 789
outbound sequences
outbound_sequence_no 789
overflow records
first_overflow_time 663
last_over_flow time 712
overflow_accesses 789
overflow_creates 790
package cache
coord_stmt_exec_time 591
last_metrics_update 712
num_coord_exec 772
num_coord_exec_with_metrics 772
pkg_cache_inserts 801
pkg_cache_lookups 803
pkg_cache_num_overflow 804
pkg_cache_size_top 805
stmt_exec_time 931
stmt_type_id 942
total_routine_invocations 1016
total_routine_non_sect_proc_time 1017
total_routine_non_sect_time 1018
total_routine_time 1018
total_section_proc_time 1027

monitor elements *(continued)*
package cache *(continued)*
total_section_time 1033
written from EVMON_FORMAT_UE_TO_TABLES table function 87
written from EVMON_FORMAT_UE_TO_XML table function 81
package_elapsed_time 790
package_id 790
package_list_count 791
package_list_exceeded 791
packages
package_name 791
package_schema 792
package_version_id 792
packet_receive_errors 793
packet_send_errors 793
packets_received 793
packets_sent 794
pages
data_object_pages 603
parallelism
degree_parallelism 616
participant_no 798
partitions
coord_partition_num 591
data_partition_id 604
partition_number 799
pass-through
passthru 800
passthru_time 799
pool_config_size 815
pool_cur_size 815
pool_id 824
pool_secondary_id 834
pool_watermark 847
prefetching
unread_prefetch_pages 1047
priv_workspace_num_overflows 862
progress_completed_units 865
progress_work_metric 867
pseudo_deletes 868
pseudo_empty_pages 868
queries
query_card_estimate 868
query_cost_estimate 869
queue_assignments_total 870
queue_size_top 870
queue_time_total 870
select_time 903
quiescer
quiescer_auth_id 871
quiescer_obj_id 871
quiescer_state 872
quiescer_ts_id 872
ranges
bottom 542
range_adjustment 872
range_container_id 873
range_end_stripe 873
range_max_extent 873
range_max_page_number 873
range_num_containers 873
range_number 874
range_offset 874
range_start_stripe 874
range_stripe_set_number 874
monitor elements *(continued)*
rebalancing
current_extent 603
rebinding
int_auto_rebinds 697
records
partial_record 798
reoptimization
stmt_value_isreopt 945
reorg_completion 877
reorg_long_tbspc_id 878
reorg_tbspc_id 881
reorganization
page_reorgs 794
reorg_current_counter 878
reorg_end 878
reorg_max_phase 879
reorg_phase 879
reorg_phase_start 880
reorg_rows_compressed 880
reorg_rows_rejected_for_compression 880
reorg_start 881
reorg_status 881
reorg_type 882
reorg_xml_regions_compressed 882
reorg_xml_regions_rejected_for_compression 883
request_exec_time_avg 884
requests
rqsts_completed_total 897
response time
delete_time 617
host_response_time 682
insert_time 696
roll-forward recovery
rf_log_num 884
rf_status 885
rf_timestamp 885
rf_type 885
rollbacks
int_rollbacks 700
rollback_sql_stmts 886
rolled_back_appl_id 887
rolled_back_participant_no 887
rolled_back_sequence_no 888
routines
routine_id 888
total_routine_user_code_time 1021
rows
int_rows_inserted 702
int_rows_updated 703
rows_deleted 889
rows_fetched 889
rows_inserted 890
rows_modified 890
rows_read 892
rows_returned 893
rows_selected 895
rows_updated 896
rows_written 897
sp_rows_selected 918
RUNSTATS utility
async_runstats 530
sync_runstats 948
sync_runstats_time 949
section_type 902
sections
priv_workspace_section_inserts 863

monitor elements *(continued)*
sections *(continued)*
priv_workspace_section_lookups 863
section_actuals 900
section_env 901
section_number 901
total_app_section_executions 993
sequences
progress_seq_num 866
sequence_no 904
servers
product_name 865
server_instance_name 905
server_platform 906
server_prdid 906
server_version 907
service levels
service_level 908
service subclasses
total_rqst_mapped_in 1022
total_rqst_mapped_out 1022
shared workspaces
shr_workspace_num_overflows 911
shr_workspace_section_inserts 911
shr_workspace_section_lookups 912
shr_workspace_size_top 913
snapshots
time_stamp 986
sorting
piped_sorts_accepted 800
piped_sorts_requested 801
post_shrthreshold_sorts 856
post_threshold_sorts 858
sort_heap_allocated 914
sort_heap_top 914
sort_overflows 915
sort_shrheap_allocated 916
sort_shrheap_top 917
total_section_sort_proc_time 1029
total_section_sort_time 1030
total_section_sorts 1032
total_sorts 1036
source_service_class_id 918
SQL communication area (SQLCA)
sqlca 920
SQL operations
elapsed_exec_time 635
SQL statements
ddl_sql_stmts 613
dynamic_sql_stmts 632
failed_sql_stmts 640
insert_sql_stmts 695
num_compilation 772
num_executions 773
select_sql_stmts 902
sql_chains 918
sql_reqs_since_commit 919
sql_stmts 919
static_sql_stmts 927
stmt_pkgcache_id 937
stmt_query_id 938
stmt_sorts 938
stmt_source_id 939
stmt_text 940
stmt_value_data 944
stmt_value_index 944
stmt_value_isnull 945
monitor elements *(continued)*
SQL statements *(continued)*
stmt_value_type 946
total_exec_time 1002
uid_sql_stmts 1046
statements
prep_time_best 861
prep_time_worst 861
stmt_first_use_time 931
stmt_history_id 932
stmt_history_list_size 932
stmt_invocation_id 703, 932
stmt_isolation 933
stmt_last_use_time 934
stmt_nest_level 768, 935
stmt_node_number 935
stmt_type 942
status
db_status 609
db2_status 606
dcs_appl_status 612
ss_status 925
storage paths
num_db_storage_paths 773
stored procedures
stored_proc_time 947
stored_procs 947
stripe sets
container_stripe_set 582
swap_page_size 948
swap_pages_in 948
swap_pages_out 948
system_auth_id 950
system_metrics 129, 163
table queues
tq_tot_send_spills 1044
table spaces
index_tbsp_id 694
long_tbsp_id 747
rebalancer_mode 966
tablespace_auto_resize_enabled 955
tablespace_content_type 956
tablespace_cur_pool_id 956
tablespace_current_size 956
tablespace_extent_size 957
tablespace_free_pages 957
tablespace_id 958
tablespace_increase_size 958
tablespace_increase_size_percent 959
tablespace_initial_size 959
tablespace_last_resize_failed 959
tablespace_last_resize_time 960
tablespace_max_size 960
tablespace_min_recovery_time 960
tablespace_name 961
tablespace_next_pool_id 962
tablespace_num_containers 962
tablespace_num_quiescers 962
tablespace_num_ranges 963
tablespace_page_size 963
tablespace_page_top 963
tablespace_pending_free_pages 964
tablespace_prefetch_size 965
tablespace_rebalancer_extents_processed 965
tablespace_rebalancer_extents_remaining 966
tablespace_rebalancer_last_extent_moved 966
tablespace_rebalancer_mode 966

monitor elements *(continued)*
table spaces *(continued)*
tablespace_rebalancer_priority 968
tablespace_rebalancer_restart_time 968
tablespace_rebalancer_start_time 968
tablespace_state 969
tablespace_state_change_object_id 971
tablespace_state_change_ts_id 971
tablespace_total_pages 971
tablespace_type 973
tablespace_usable_pages 973
tablespace_used_pages 974
tablespace_using_auto_storage 974
tbsp_auto_resize_enabled 955
tbsp_content_type 956
tbsp_cur_pool_id 956
tbsp_current_size 956
tbsp_extent_size 957
tbsp_free_pages 957
tbsp_id 958
tbsp_increase_size 958
tbsp_increase_size_percent 959
tbsp_initial_size 959
tbsp_last_resize_failed 959
tbsp_last_resize_time 960
tbsp_max_page_top 975
tbsp_max_size 960
tbsp_min_recovery_time 960
tbsp_name 961
tbsp_next_pool_id 962
tbsp_num_containers 962
tbsp_num_quiescers 962
tbsp_num_ranges 963
tbsp_page_size 963
tbsp_page_top 963
tbsp_pending_free_pages 964
tbsp_prefetch_size 965
tbsp_rebalancer_extents_processed 965
tbsp_rebalancer_extents_remaining 966
tbsp_rebalancer_last_extent_moved 966
tbsp_rebalancer_mode 966
tbsp_rebalancer_priority 968
tbsp_rebalancer_restart_time 968
tbsp_rebalancer_start_time 968
tbsp_state 969
tbsp_state_change_object_id 971
tbsp_state_change_ts_id 971
tbsp_total_pages 971
tbsp_trackmod_state 972
tbsp_type 973
tbsp_usable_pages 973
tbsp_used_pages 974
tbsp_using_auto_storage 974
ts_name 1046
tables
tab_file_id 951
tab_type 951
table_file_id 951
table_name 952
table_scans 953
table_schema 953
table_type 954
TCP/IP
tcpip_sends_total 980
territory_code 981
thresholds
num_lw_thresh_exceeded 778
monitor elements *(continued)*
thresholds *(continued)*
num_threshold_violations 780
thresh_violations 982
threshold_action 983
threshold_domain 983
threshold_maxvalue 984
threshold_name 984
threshold_predicate 984
threshold_queuesize 985
thresholdid 985
time
prefetch_wait_time 860
prep_time 860
progress_start_time 867
ss_exec_time 924
stmt_elapsed_time 930
time_completed 985
time_created 986
time_of_violation 986
time_started 986
total_sort_time 1035
time spent
examples of usage 248
hierarchy 241, 413
overview 239
ranking 248
viewing time spent during SQL statement execution 252
viewing time spent in activities 252
viewing time spent in the system 248
time spent metrics
total_act_wait_time 989
time stamps
activate_timestamp 501
db_conn_time 607
db2start_time 606
last_backup 711
last_reset 714
lock_wait_end_time 733
lock_wait_start_time 734
message_time 768
statistics_timestamp 928
status_change_time 930
stmt_start 939
stmt_stop 939
time zones
time_zone_disp 987
tokens
consistency_token 580
corr_token 591
total_app_commits 990
total_app_rollbacks 991
total_bytes_received 995
total_bytes_sent 995
total_commit_proc_time 995
total_commit_time 996
total_compilations 997
total_compile_proc_time 998
total_compile_time 999
total_hash_loops 1003
total_implicit_compilations 1003
total_implicit_compile_proc_time 1005
total_implicit_compile_time 1006
total_load_proc_time 1006
total_load_time 1007
total_loads 1008

monitor elements *(continued)*
total_reorg_proc_time 1011
total_reorg_time 1012
total_reorgs 1013
total_rollback_proc_time 1014
total_rollback_time 1015
total_runstats 1024
total_runstats_proc_time 1025
total_runstats_time 1026
tq_cur_send_spills 1042
tq_id_waiting_on 1042
tq_max_send_spills 1042
tq_node_waited_for 1043
tq_rows_read 1043
tq_rows_written 1044
tq_wait_for_any 1045
transactions
client_acctng 554
client_userid 562
client_wrkstnname 563
num_indoubt_trans 775
tpmon_acc_str 1039
tpmon_client_userid 1041
tpmon_client_wkstn 1041
xid monitor 1068
units of work (UOW)
completion_status 566
parent_uow_id 797
prev_uow_stop_time 861
progress_total_units 867
uow_comp_status 1047
uow_elapsed_time 1048
uow_id 1048
uow_start_time 1050
uow_status 1051
uow_stop_time 1051
written from EVMON_FORMAT_UE_TO_TABLES table function 119
written from EVMON_FORMAT_UE_TO_XML table function 107
updates
update_sql_stmts 1053
utilities
utility_dbname 1054
utility_description 1054
utility_id 1054
utility_invoker_type 1055
utility_priority 1055
utility_start_time 1055
utility_state 1055
utility_type 1056
valid 1056
viewing as rows in a table 18
virtual_mem_free 1057
virtual_mem_reserved 1058
virtual_mem_total 1058
wait time 241, 413
wait times
diaglog_write_wait_time 618
lock_wait_time_top 736
prefetch_wait_time 860
total_wait_time 1038
watermarks
act_cpu_time_top 496
act_rows_read_top 499
concurrent_act_top 568
concurrent_connection_top 568
monitor elements *(continued)*
watermarks *(continued)*
concurrent_wlo_act_top 569
concurrent_wlo_top 569
coord_act_lifetime_top 587
cost_estimate_top 592
lock_wait_time_top 736
rows_returned_top 895
temp_tablespace_top 981
uow_total_time_top 1052
workload management
total queue assignments 1059
total queue time 1060
wl_work_action_set_id 1058
wl_work_class_id 1059
workloads
wlo_completed_total 1062
workload_id 1063
workload_name 1064
workload_occurrence_id 1065
workload_occurrence_state 1066
xda_object_l_pages - XML storage object (XDA) data logical pages 1068
XQuery
xquery_stmts 1068
monitor heap health indicator 357
monitor switches
details 283
setting
client applications 287
CLP 285
monitoring
API request types 270
buffer pool efficiency 372
CLP commands 266
data partitions 374
database activities 367, 373
database events
configuring event monitors 197
event types 23
sample output 217
databases 3
db2top command 278
events
locking 35
units of work 98
extent movement status
table functions 8
FCM (Fast Communications Manager)
table functions 8
health monitor 299, 307
locking
event monitors 35
table functions 8
memory components 364
monitor data returned in XML documents 9
package cache events 68
runtime rollback process 373
snapshot access
snapshot table functions in SQL queries 261
SYSMON authority 255
snapshot capture methods
client applications 268
CLP 265
SNAP_WRITE_FILE stored procedure 258
snapshot administrative views 256
snapshot table functions 256

monitoring *(continued)*
snapshot capture methods *(continued)*
snapshot table functions in SQL queries 261
SQL 264
system catalog views directly
last referenced date 237
table functions 5
unformatted event table 27
unit of work events 94
monitoring data 223
monitoring information 227
MONREPORT 227, 230, 232, 233, 234
MONREPORT reports 223
most recent response time for connect monitor element 567

N

names
monitor elements
db_name 608
dcs_db_name 612
service_subclass_name 908
service_superclass_name 909
work_action_set_name 1062
work_class_name 1063
network time
monitor elements
max_network_time_1_ms 759
max_network_time_100_ms 758
max_network_time_16_ms 758
max_network_time_4_ms 759
max_network_time_500_ms 760
max_network_time_gt500_ms 760
network_time_bottom 769
network_time_top 769
nicknames
health indicator 358
monitor elements
create_nickname 600
create_nickname_time 601
nodes
monitor elements
coord_node 590
node_number 770
num_nodes_in_db2_instance 780
ss_node_number 925
notices 1085
num_indoubt_trans element 775
num_transmissions element 781
num_transmissions_group element 781
numbers
monitor elements
progress_list_cur_seq_num 866
ss_number 925

O

objects
performance (Windows) 387
OLAP
monitor elements
active_olap_funcs 501
olap_func_overflows 782
post_threshold_olap_funcs 858
total_olap_funcs 1010
operation monitor element 935
operations
monitor elements
direct_read_reqs 620
direct_read_time 622
direct_reads 624
direct_write_reqs 626
direct_write_time 627
direct_writes 629
stmt_operation 935
optimization
monitor elements
stmt_value_isreopt 945
ordering DB2 books 1076
outbound bytes received
monitor elements
max_data_received_1024 748
max_data_received_128 749
max_data_received_16384 749
max_data_received_2048 750
max_data_received_256 750
max_data_received_31999 751
max_data_received_4096 751
max_data_received_512 751
max_data_received_64000 752
max_data_received_8192 752
max_data_received_gt64000 753
outbound_bytes_received 786
outbound_bytes_received_bottom 787
outbound_bytes_received_top 787
outbound bytes sent
monitor elements
max_data_sent_1024 753
max_data_sent_128 753
max_data_sent_16384 754
max_data_sent_2048 754
max_data_sent_256 755
max_data_sent_31999 755
max_data_sent_4096 756
max_data_sent_512 756
max_data_sent_64000 757
max_data_sent_8192 757
max_data_sent_gt64000 757
outbound_bytes_sent 788
outbound_bytes_sent_bottom 788
outbound_bytes_sent_top 788
outbound communication
monitor elements
outbound_appl_id 786
outbound_comm_address 788
outbound_comm_protocol 789
outbound_sequence_no 789
overflow records
event monitors 204
monitor elements
first_overflow_time 663
last_overflow_time 712
overflow_accesses 789
overflow_creates 790
overview
MONREPORT 223

P

package cache
db.pkgcache_hitratio health indicator 356
monitor elements
pkg_cache_inserts 801

package cache *(continued)*
monitor elements *(continued)*
pkg_cache_lookups 803
pkg_cache_num_overflow 804
pkg_cache_size_top 805
package cache event monitor
monitor data returned in XML documents 9
usage example
improving database performance 76
tuning statements 73
package cache event monitor reports 71
package listing
unit of work event monitor 103
packages
monitor elements
package_name 791
package_schema 792
package_version_id 792
stmt_pkgcache_id 937
pages
bp_pages_left_to_remove monitor element 544
data_object_pages monitor element 603
removing 544
parallelism
monitor elements
degree_parallelism 616
partial_record monitor element 798
partition_number monitor element 799
partitioned database environments
event monitoring 214
global snapshots 275
monitor elements
coord_partition_num 591
partitioned tables
reorganizing 374
pass-through monitor elements
passthru_time 799
passthrus 800
pattern
slow down 233
performance
information
displaying 387
enabling remote access 386
remote databases 388
resetting values 388
Windows
monitoring tools 385
Performance Monitor objects 387
performance tuning
db2advis
creating input file with the package cache event monitor 76
identifying costly statements from the package cache 73
time-spent monitor elements 239
pipe event monitors
creating 213
formatting output from command line 217
named pipe management 214
piped_sorts_accepted monitor element 800
piped_sorts_requested monitor element 801
post_shrthreshold_sorts monitor element 856
prefetching
unread_prefetch_page monitor elements 1047
priv_workspace_num_overflows monitor element 862
priv_workspace_section_inserts monitor element 863
priv_workspace_section_lookups monitor element 863
priv_workspace_size_top monitor element 864
problem determination
information available 1082
tutorials 1082
processes
monitor elements
agent_pid 509
processor utilization
monitor elements
cpu_idle 593
cpu_iowait 594
cpu_system 596
cpu_usage_total 597
cpu_user 598
progress_description monitor element 865
progress_seq_num monitor element 866
progress_start_time monitor element 867
progress_work_metric monitor element 867

Q

queries
monitor elements
query_card_estimate 868
query_cost_estimate 869
queue_assignments_total 870
queue_size_top 870
queue_time_total 870
select_time 903
quiescer
monitor elements
quiescer_auth_id 871
quiescer_obj_id 871
quiescer_state 872
quiescer_ts_id 872

R

range adjustment monitor element 872
range container monitor element 873
range number monitor element 874
range offset monitor element 874
range_num_containers monitor element 873
ranges
monitor elements
bottom 542
range_adjustment 872
range_container_id 873
range_end_stripe 873
range_max_extent 873
range_max_page_number 873
range_num_containers 873
range_number 874
range_offset 874
range_start_stripe 874
range_stripe_set_number 874
ranking monitor elements 18
real-time statistics
monitor elements
stats_fabricate_time 929
stats_fabrications 929
rebalancing
monitor elements
current_extent 603
rebalancer_mode 966
tablespace_rebalancer_extents_processed 965

rebalancing *(continued)*
monitor elements *(continued)*
tablespace_rebalancer_extents_remaining 966
tablespace_rebalancer_last_extent_moved 966
tablespace_rebalancer_mode 966
tablespace_rebalancer_priority 968
tablespace_rebalancer_restart_time 968
tablespace_rebalancer_start_time 968
tbsp_rebalancer_extents_processed 965
tbsp_rebalancer_extents_remaining 966
tbsp_rebalancer_last_extent_moved 966
tbsp_rebalancer_mode 966
tbsp_rebalancer_priority 968
tbsp_rebalancer_restart_time 968
tbsp_rebalancer_start_time 968
rebinding
monitor elements
int_auto_rebinds 697
records
monitor elements
partial_record 798
recovery
monitor elements
log_to_redo_for_recovery 745
remote databases
performance information 388
reoptimization monitor elements
stmt_value_isreopt 945
reorg_index_id monitor element 878
reorganization
health indicators
db.tb_reorg_req 349
monitor elements
page_reorgs 794
reorg_current_counter 878
reorg_max_counter 879
reorg_max_phase 879
reorg_phase 879
reorg_phase_start 880
reorg_rows_compressed 880
reorg_rows_rejected_for_compression 880
reorg_start 881
reorg_status 881
reorg_type 882
reorganize phase monitor element 879
reports
deadlock 37
lock timeouts 37
lock waits 37
package cache 71
units of work 96
request identifier for sql statement monitor element 919
request monitor elements
overview 407
rqsts_completed_total 897
response times
monitor elements
delete_time 617
host_response_time 682
insert_time 696
review work 233
roll-forward recovery
monitor elements
rf_log_num 884
rf_status 885
rf_timestamp 885
rf_type 885
roll-forward recovery *(continued)*
monitor elements *(continued)*
tablespace_min_recovery_time 960
tbsp_min_recovery_time 960
ts_name 1046
rollbacks
monitor elements
int_deadlock_rollbacks 699
int_rollbacks 700
rf_status 885
rollback_sql_stmts 886
rolled_back_agent_id 887
rolled_back_appl_id 887
rolled_back_participant_no 887
rolled_back_sequence_no 888
monitoring progress 373
routines
monitor elements
routine_id 888
row-based formatting functions 18
rows
monitor elements
int_rows_inserted 702
int_rows_updated 703
rows_deleted 889
rows_fetched 889
rows_inserted 890
rows_modified 890
rows_read 892
rows_returned 893
rows_returned_top 895
rows_selected 895
rows_updated 896
rows_written 897
sp_rows_selected 918
rows compressed monitor element 880
rows rejected for compression monitor element 880
rows returned by stored procedures monitor element 918
rows selected monitor element 895
RUNSTATS utility
monitor elements
async_runstats 530
sync_runstats 948
sync_runstats_time 949

S

schemas
table_schema monitor element 953
sections
monitor elements
appl_section_inserts 525
appl_section_lookups 525
priv_workspace_section_inserts 863
priv_workspace_section_lookups 863
section_env 901
section_number 901
select SQL statements executed monitor element 902
self-describing data streams
database system monitor 291
event monitors 218
snapshot monitor 275
system monitor switches 288
sequences
monitor elements
progress_seq_num 866
sequence_no 904

sequences *(continued)*
monitor elements *(continued)*
sequence_no_holding_lk 904
servers
monitor elements
product_name 865
server_instance_name 905
server_platform 906
server_prdid 906
server_version 907
service-level information
service_level monitor element 908
session authorization IDs
monitor element 910
shared workspaces
health indicators
db.shrworkspace_hitratio 357
monitor elements
shr_workspace_num_overflows 911
shr_workspace_section_inserts 911
shr_workspace_section_lookups 912
shr_workspace_size_top 913
slow down 234
slow response 227
snapshot monitoring
administrative views 370
API request types 270
capturing snapshots
to file 258
using SQL with file access 261
CLP commands 266
data partitions 374
interpreting output for data partitions 374
making snapshot data available for all users 258
methods
client applications 268
CLP 265
SNAP_WRITE_FILE stored procedure 258
SQL 264
SQL with direct access 256
output
samples 272
self-describing data streams 275
overview 255
partitioned database systems 275
request types 266
SQL table functions 261
subsections 274
snapshot time monitor element 986
snapshots
monitor elements
time_stamp 986
sort share heap currently allocated monitor element 916
sort share heap high watermark monitor element 917
sorting
health indicators
db2.sort_privmem_util 344
monitor elements
active_sorts 502
db.spilled_sorts 346
piped_sorts_accepted 800
piped_sorts_requested 801
post_shrthreshold_sorts 856
post_threshold_sorts 858
sort_heap_allocated 914
sort_heap_top 914
sort_overflows 915
sorting *(continued)*
monitor elements *(continued)*
sort_shrheap_allocated 916
sort_shrheap_top 917
total_sorts 1036
SQL
operations
elapsed_exec_time element monitor element 635
table functions
capturing health snapshots 309
health monitor 361
SQL performance
slowdown 232
SQL requests since last commit monitor element 919
SQL slow down 230
SQL slowdown 230
SQL statement
stopped 230
SQL statements 234
help
displaying 1077
monitor elements
ddl_sql_stmts 613
dynamic_sql_stmts 632
failed_sql_stmts 640
insert_sql_stmts 695
num_compilation 772
num_executions 773
prep_time_best 861
prep_time_worst 861
select_sql_stmts 902
sql_chains 918
sql_reqs_since_commit 919
sql_stmts 919
static_sql_stmts 927
stmt_first_use_time 931
stmt_history_id 932
stmt_history_list_size 932
stmt_invocation_id 703, 932
stmt_isolation 933
stmt_last_use_time 934
stmt_nest_level 768, 935
stmt_node_number 935
stmt_pkgcache_id 937
stmt_query_id 938
stmt_sorts 938
stmt_source_id 939
stmt_text 940
stmt_type 942
stmt_value_data 944
stmt_value_index 944
stmt_value_isnull 945
stmt_value_type 946
total_exec_time 1002
uid_sql_stmts 1046
sql_chains element 918
sql_stmts element 919
SQLCA
monitor elements
sqlca 920
SQLTEMPSPACE activity threshold
monitor elements
sqltempspace_threshold_id 923
start stripe monitor element 874
statement best preparation time monitor element 861

statement concentrator
monitor elements
eff_stmt_txt 633
statement first use time monitor element 931
statement history identifier monitor element 932
statement history list size monitor element 932
statement invocation identifier monitor element 703, 932
statement isolation monitor element 933
statement last use time monitor element 934
statement nesting level monitor element 768, 935
statement node monitor element 935
statement operation monitor element 935
statement query identifier monitor element 938
statement sorts monitor element 938
statement source identifier monitor element 939
statement type monitor element 942
statement worst preparation time monitor element 861
STATEMENTS event type
overview 23
states
health indicators
db.alert_state 349
db.db_op_status 348
db2.db2_op_status 347
ts.ts_op_status 343
static SQL statements attempted monitor element 927
statistics
collection
health indicators 350
statistics event monitor
monitor data returned in XML documents 9
status
monitor elements
appl_status 525
db_status 609
db2_status 606
dcs_appl_status 612
ss_status 925
stmt_operation element 935
storage paths
monitor elements
num_db_storage_paths 773
stored procedure time monitor element 947
stored procedures
monitor elements
stored_proc_time 947
stored_procs 947
stored procedures monitor element 947
stripe set number monitor element 874
stripe sets
monitor elements
container_stripe_set 582
subsection execution elapsed time monitor element 924
subsection node number monitor element 925
subsection number monitor element 925
subsection status monitor element 925
subsections
snapshots 274
SYSMON (system monitor) authority
details 255
System Monitor Guide and Reference
overview xix
system monitor switches
details 283
self-describing data streams 288
setting
client applications 287
system monitor switches *(continued)*
setting *(continued)*
CLP 285
types 283

T

table event monitors
creating 201
table management 204
table functions
monitor 5
monitoring
activities 6
data objects 7
extent movement 8
FCM (Fast Communications Manager) 8
locking 8
memory 8
miscellaneous 8
system information 5
table queues
monitor elements
tq_cur_send_spills 1042
tq_id_waiting_on 1042
tq_max_send_spills 1042
tq_node_waited_for 1043
tq_rows_read 1043
tq_rows_written 1044
tq_tot_send_spills 1044
tq_wait_for_any 1045
table reorganization
monitor elements
reorg_end 878
reorg_xml_regions_compressed 882
reorg_xml_regions_rejected_for_compression 883
table reorganize attribute flag 882
table reorganize completion flag 877
table reorganize end time 878
table reorganize phase start time 880
table reorganize start time 881
table reorganize status 881
table reorganize attribute flag monitor element 882
table reorganize completion flag monitor element 877
table reorganize end time monitor element 878
table reorganize phase start time monitor element 880
table reorganize start time monitor element 881
table reorganize status monitor element 881
table spaces
health indicators
ts.ts_auto_resize_status 340
ts.ts_op_status 343
ts.ts_util 341
ts.ts_util_auto_resize 341
tsc.tscont_op_status 344
tsc.utilization 342
monitor elements
bp_tbsp_use_count 544
index_tbsp_id 694
long_tbsp_id 747
quiescer_ts_id 872
rebalancer_mode 966
reorg_long_tbspc_id 878
reorg_tbspc_id 881
tablespace_auto_resize_enabled 955
tablespace_content_type 956
tablespace_cur_pool_id 956

table spaces *(continued)*
monitor elements *(continued)*
tablespace_current_size 956
tablespace_extent_size 957
tablespace_free_pages 957
tablespace_id 958
tablespace_increase_size 958
tablespace_increase_size_percent 959
tablespace_initial_size 959
tablespace_last_resize_failed 959
tablespace_last_resize_time 960
tablespace_max_size 960
tablespace_min_recovery_time 960
tablespace_name 961
tablespace_next_pool_id 962
tablespace_num_containers 962
tablespace_num_quiescers 962
tablespace_num_ranges 963
tablespace_page_size 963
tablespace_page_top 963
tablespace_pending_free_pages 964
tablespace_prefetch_size 965
tablespace_rebalancer_extents_processed 965
tablespace_rebalancer_extents_remaining 966
tablespace_rebalancer_last_extent_moved 966
tablespace_rebalancer_mode 966
tablespace_rebalancer_priority 968
tablespace_rebalancer_restart_time 968
tablespace_rebalancer_start_time 968
tablespace_state 969
tablespace_state_change_object_id 971
tablespace_state_change_ts_id 971
tablespace_total_pages 971
tablespace_type 973
tablespace_usable_pages 973
tablespace_used_pages 974
tablespace_using_auto_storage 974
tbsp_auto_resize_enabled 955
tbsp_content_type 956
tbsp_cur_pool_id 956
tbsp_current_size 956
tbsp_extent_size 957
tbsp_free_pages 957
tbsp_id 958
tbsp_increase_size 958
tbsp_increase_size_percent 959
tbsp_initial_size 959
tbsp_last_resize_failed 959
tbsp_last_resize_time 960
tbsp_max_page_top 975
tbsp_max_size 960
tbsp_min_recovery_time 960
tbsp_name 961
tbsp_next_pool_id 962
tbsp_num_containers 962
tbsp_num_quiescers 962
tbsp_num_ranges 963
tbsp_page_size 963
tbsp_page_top 963
tbsp_pending_free_pages 964
tbsp_prefetch_size 965
tbsp_rebalancer_extents_processed 965
tbsp_rebalancer_extents_remaining 966
tbsp_rebalancer_last_extent_moved 966
tbsp_rebalancer_mode 966
tbsp_rebalancer_priority 968
tbsp_rebalancer_restart_time 968
table spaces *(continued)*
monitor elements *(continued)*
tbsp_rebalancer_start_time 968
tbsp_state 969
tbsp_state_change_object_id 971
tbsp_state_change_ts_id 971
tbsp_total_pages 971
tbsp_trackmod_state 972
tbsp_type 973
tbsp_usable_pages 973
tbsp_used_pages 974
tbsp_using_auto_storage 974
ts_name 1046
tables
monitor elements
tab_file_id 951
tab_type 951
table_file_id 951
table_name 952
table_scans 953
table_schema 953
table_type 954
TABLES event type
overview 23
TABLESPACES event type 23
target tables
event monitors 204
TCP/IP
monitor elements
tcpip_sends_total 980
terms and conditions
publications 1082
territory codes
monitor elements
territory_code 981
threads
monitor elements
agent_pid 509
thresholds
health indicators 299, 333
monitor elements
num_threshold_violations 780
sqltempspace_threshold_id 923
thresh_violations 982
threshold_action 983
threshold_domain 983
threshold_maxvalue 984
threshold_name 984
threshold_predicate 984
threshold_queuesize 985
thresholdid 985
time
monitor elements
prefetch_wait_time 860
prep_time 860
progress_start_time 867
ss_exec_time 924
stmt_elapsed_time 930
time_completed 985
time_created 986
time_of_violation 986
time_started 986
total_sort_time 1035
time spent
monitor elements
examples 248
hierarchy 241, 413

time spent *(continued)*
monitor elements *(continued)*
overview 239
viewing as rows in a table 18
viewing
across the system 248
during SQL statement execution 252
time stamps
monitor elements
activate_timestamp 501
db_conn_time 607
db2start_time 606
last_backup 711
last_reset 714
llock_wait_end_time 733
lock_wait_start_time 734
message_time 768
prev_uow_stop_time 861
statistics_timestamp 928
status_change_time 930
stmt_start 939
stmt_stop 939
uow_start_time 1050
uow_stop_time 1051
time waited for prefetch monitor element 860
time zone displacement monitor element 987
time zones
time_zone_disp element monitor element 987
tokens
monitor elements
consistency_token 580
corr_token 591
top statement 232
total hash loops monitor element 1003
total log available monitor element 1009
total log space used monitor element 1010
total number of pages in object monitor element 879
total progress work units monitor element 867
total sort time monitor element 1035
total sorts monitor element 1036
transaction processing monitors
monitor elements
client_acctng 554
client_applname 555
client_userid 562
client_wrkstnname 563
tpmon_acc_str 1039
tpmon_client_app 1040
tpmon_client_userid 1041
tpmon_client_wkstn 1041
transactions
monitor elements
num_indoubt_trans 775
xid 1068
TRANSACTIONS event type 23
troubleshoot 230
slow down 227
slowdown 227
SQL 223, 227
SQL slowdown 232
troubleshooting
online information 1082
SQL slowdown 233
tutorials 1082
troubleshootingproblem
performance 227
SQL 227
troubleshootperformance
slowdown 234
SQL 234
tutorials
list 1081
problem determination 1082
troubleshooting 1082
Visual Explain 1081
type at monitored (server) node monitor element 905

U

unformatted event tables
db2evmonfmt Java-based tool for parsing data 30
unit of work event monitor
collecting data 96
monitor data returned in XML documents 9
usage example 98
unit of work events
monitoring 94
units of work (UOW)
monitor elements
completion_status 566
parent_uow_id 797
prev_uow_stop_time 861
progress_total_units 867
uow_comp_status 1047
uow_elapsed_time 1048
uow_id 1048
uow_lock_wait_time 1049
uow_log_space_used 1049
uow_start_time 1050
uow_status 1051
uow_stop_time 1051
update response time monitor element 1053
update_time element 1053
update/insert/delete SQL statements executed monitor element 1046
updates
DB2 Information Center 1079, 1080
monitor elements
update_sql_stmts 1053
updates monitor element 1053
user authorization level monitor element 538
utilities
monitor elements
utility_dbname 1054
utility_description 1054
utility_id 1054
utility_invoker_type 1055
utility_priority 1055
utility_start_time 1055
utility_state 1055
utility_type 1056

V

value data monitor element 944
value has null value monitor element 945
value index monitor element 944
value type monitor element 946
version monitor element 1057

W

wait times
  monitor elements
    FCM (Fast Communications Manager) 246
    overview 241, 413
    total_wait_time 1038
  viewing
    activity-level examples 252
    system-level examples 248
watermark monitor elements
  act_cpu_time_top 496
  act_rows_read_top 499
  concurrent_act_top 568
  concurrent_connection_top 568
  concurrent_wlo_act_top 569
  concurrent_wlo_top 569
  coord_act_lifetime_top 587
  cost_estimate_top 592
  lock_wait_time_top 736
  rows_returned_top 895
  temp_tablespace_top 981
  uow_total_time_top 1052
Windows
  Performance Monitor
    overview 385
    registering DB2 386
Windows Management Instrumentation (WMI)
  DB2 database system integration 384
  details 383
workloads
  monitor elements
    wlo_completed_total 1062
    workload_id 1063
    workload_name 1064
    workload_occurrence_id 1065
    workload_occurrence_state 1066
write-to-table event monitors
  buffering 212

X

XDA Object Pages monitor element 1067
xda_object_pages monitor element 1067
XML
  monitor elements
    formatting 18
    overview 9
XML documents
  monitor elements 9
XMLTABLE table function
  comparison with MON_FORMAT_ table functions 14
xquery_stmts monitor element 1068

Printed in USA

SC27-2458-03

Spine information:

DB2 for Linux, UNIX, and Windows **Version 9 Release 7**

Database Monitoring Guide and Reference