



IBM DB2 Universal Database

Command Reference

Version 5.2



IBM DB2 Universal Database

Command Reference

Version 5.2

Before using this information and the product it supports, be sure to read the general information under Appendix E, "Notices" on page 487.

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.

Order publications through your IBM representative or the IBM branch office serving your locality or by calling 1-800-879-2755 in U.S. or 1-800-IBM-4YOU in Canada.

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 International Business Machines Corporation 1993, 1998. All rights reserved.**

Note to U.S. Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Contents

About This Book	ix
Who Should Use this Book	ix
How this Book is Structured	ix
Chapter 1. System Commands	1
How the Command Descriptions are Organized	1
db2admin - DB2 Administration Server	3
db2adutl - Work with ADSM Archived Images	5
db2audit - Audit Facility Administrator Tool	6
db2atld - Autoloader	7
db2batch - Benchmark Tool	8
db2bfd - Bind File Description Tool	13
db2cc - Start Control Center	14
db2cidmg - Remote Database Migration	16
db2ckmig - Database Pre-migration Tool	17
db2cli - DB2 Interactive CLI	18
db2cmd - Open DB2 Command Window	19
db2drdat - DRDA Trace	20
db2empfa - Enable Multi-page File Allocation	22
db2eva - Event Analyzer	23
db2evmon - Event Monitor Productivity Tool	25
db2exfmt - Explain Table Format Tool	26
db2expln - DB2 SQL Explain Tool	27
db2flsn - Find Log Sequence Number	28
db2gov - DB2 Governor	30
db2govlg - DB2 Governor Log Query	31
db2icrt - Create Instance	32
db2idrop - Remove Instance	33
db2ilist - List Instances	34
db2imigr - Migrate Instance	35
db2ipxad - Get IPX/SPX Internetwork Address	36
db2iupdt - Update Instances	37
db2look - DB2 Statistics Extraction Tool	38
db2migdr - Local Database Directory Migration	41
db2mscs - Set up Windows NT Failover Utility	42
db2profc - DB2 SQLJ Profile Customizer	43
db2profp - DB2 SQLJ Profile Printer	45
db2rbind - Rebind all Packages	47
db2sampl - Create Sample Database	48
db2set - DB2 Profile Registry Command	50
db2split - Data Declustering Tool	53
db2sql92 - SQL92 Compliant SQL Statement Processor	54
db2start - Start DB2	57
db2stop - Stop DB2	58
db2tbst - Get Tablespace State	59

db2trc - Trace	60
db2uiddl - Prepare Unique Index Conversion to V5 Semantics	63
db2untag - Release Container Tag	64
db2upd52 - Update Catalog to Support Version 5.2	66
db2vexp - Dynamic Visual Explain	67
Chapter 2. Command Line Processor (CLP)	69
Command Line Processor Invocation and Options	69
Using the Command Line Processor	78
Chapter 3. CLP Commands	83
DB2 CLP Commands	83
ACTIVATE DATABASE	87
ADD NODE	89
ATTACH	91
BACKUP DATABASE	93
BIND	98
CATALOG APPC NODE	111
CATALOG APPCLU NODE	114
CATALOG APPN NODE	116
CATALOG DATABASE	118
CATALOG DCS DATABASE	121
CATALOG GLOBAL DATABASE	124
CATALOG IPX/SPX NODE	126
CATALOG LOCAL NODE	129
CATALOG NAMED PIPE NODE	131
CATALOG NETBIOS NODE	133
CATALOG ODBC DATA SOURCE	135
CATALOG TCP/IP NODE	136
CHANGE DATABASE COMMENT	139
CHANGE ISOLATION LEVEL	141
CREATE DATABASE	143
DEACTIVATE DATABASE	149
DEREGISTER	151
DESCRIBE	152
DETACH	156
DROP DATABASE	157
DROP NODE VERIFY	159
ECHO	160
EXPORT	161
FORCE APPLICATION	169
GET ADMIN CONFIGURATION	171
GET AUTHORIZATIONS	174
GET CLI CONFIGURATION	176
GET CONNECTION STATE	178
GET DATABASE CONFIGURATION	179
GET DATABASE MANAGER CONFIGURATION	188
GET DATABASE MANAGER MONITOR SWITCHES	198

GET INSTANCE	200
GET MONITOR SWITCHES	201
GET SNAPSHOT	203
HELP	217
IMPORT	219
INITIALIZE TAPE	232
INVOKE STORED PROCEDURE	233
LIST ACTIVE DATABASES	235
LIST APPLICATIONS	236
LIST COMMAND OPTIONS	238
LIST DATABASE DIRECTORY	240
LIST DCS APPLICATIONS	244
LIST DCS DIRECTORY	247
LIST DRDA INDOUBT TRANSACTIONS	249
LIST HISTORY	251
LIST INDOUBT TRANSACTIONS	254
LIST NODE DIRECTORY	258
LIST NODEGROUPS	261
LIST NODES	263
LIST ODBC DATA SOURCES	264
LIST PACKAGES/TABLES	266
LIST TABLESPACE CONTAINERS	269
LIST TABLESPACES	271
LOAD	276
LOAD QUERY	301
MIGRATE DATABASE	303
PRECOMPILE PROGRAM	305
PRUNE HISTORY	325
QUERY CLIENT	326
QUIESCE TABLESPACES FOR TABLE	328
QUIT	331
REBIND	332
RECONCILE	335
REDISTRIBUTE NODEGROUP	337
REGISTER	340
REORGANIZE TABLE	342
REORGCHK	345
RESET ADMIN CONFIGURATION	352
RESET DATABASE CONFIGURATION	354
RESET DATABASE MANAGER CONFIGURATION	356
RESET MONITOR	358
RESTART DATABASE	360
RESTORE DATABASE	362
REWIND TAPE	369
ROLLFORWARD DATABASE	370
RUNSTATS	378
SET CLIENT	382
SET RUNTIME DEGREE	385

SET TABLESPACE CONTAINERS	387
SET TAPE POSITION	389
START DATABASE MANAGER	390
STOP DATABASE MANAGER	395
TERMINATE	398
UNCATALOG DATABASE	399
UNCATALOG DCS DATABASE	401
UNCATALOG NODE	402
UNCATALOG ODBC DATA SOURCE	404
UPDATE ADMIN CONFIGURATION	405
UPDATE CLI CONFIGURATION	407
UPDATE COMMAND OPTIONS	409
UPDATE DATABASE CONFIGURATION	411
UPDATE DATABASE MANAGER CONFIGURATION	413
UPDATE MONITOR SWITCHES	415
UPDATE RECOVERY HISTORY FILE	417
Chapter 4. Using Command Line SQL Statements	419
Appendix A. How to Read the Syntax Diagrams	427
Appendix B. Naming Conventions	431
Appendix C. IMPORT/EXPORT/LOAD Utility File Formats	433
Delimited ASCII (DEL) File Format	433
Sample DEL File	434
DEL Data Type Descriptions	436
PC Version of IXF File Format	438
PC/IXF Record Types	440
PC/IXF Data Types	449
PC/IXF Data Type Descriptions	455
General Rules Governing PC/IXF File Import into Databases	459
Data Type-Specific Rules Governing PC/IXF File Import into Databases	461
FORCEIN Option	463
Differences between Version 1 PC/IXF and Version 0 System/370 IXF	470
Non-delimited ASCII (ASC) File Format	471
Sample ASC File	472
ASC Data Type Descriptions	472
Appendix D. How the DB2 Library Is Structured	477
SmartGuides	477
Online Help	478
DB2 Books	479
Viewing Online Books	483
Searching Online Books	484
Printing the PostScript Books	484
Ordering the Printed DB2 Books	485
Information Center	486

Appendix E. Notices 487
Trademarks 487
Trademarks of Other Companies 488

Index 489

Contacting IBM 491

About This Book

This book provides information about the use of system commands and the IBM DB2 Universal Database command line processor (CLP) to execute database administrative functions.

Who Should Use this Book

It is assumed that the reader has an understanding of database administration and a knowledge of Structured Query Language (SQL).

How this Book is Structured

This book provides the reference information needed to use the CLP.

The following topics are covered:

- | | |
|------------|---|
| Chapter 1 | Describes the commands that can be entered at an operating system command prompt or in a shell script to access the database manager. |
| Chapter 2 | Explains how to invoke and use the command line processor, and describes the CLP options. |
| Chapter 3 | Provides a description of all database manager commands. |
| Chapter 4 | Provides information on how to use SQL statements from the command line. |
| Appendix A | Explains the conventions used in syntax diagrams. |
| Appendix B | Explains the conventions used to name objects such as databases and tables. |
| Appendix C | Describes external file formats supported by the database manager import, export, and load utilities. |

Chapter 1. System Commands

This chapter provides information about the commands that can be entered at an operating system command prompt, or in a shell script, to access and maintain the database manager.

Note: Slashes (/) in directory paths are specific to UNIX based systems, and are equivalent to back slashes (\) in directory paths on OS/2 and Windows operating systems.

How the Command Descriptions are Organized

A short description of each command precedes some or all of the following subsections.

Scope

The command's scope of operation within the instance. In a single-node system, the scope is that single node only. In a multi-node system, it is the collection of all logical nodes defined in the node configuration file, `db2nodes.cfg`.

Authorization

The authority required to successfully invoke the command.

Required Connection

One of the following: database, instance, none, or establishes a connection. Indicates whether the function requires a database connection, an instance attachment, or no connection to operate successfully. An explicit connection to the database or attachment to the instance may be required before a particular command can be issued. Commands that require a database connection or an instance attachment can be executed either locally or remotely. Those that require neither cannot be executed remotely; when issued at the client, they affect the client environment only. For information about database connections and instance attachments, see the *Administration Guide*.

Command Syntax

For information about syntax diagrams, see Appendix A, "How to Read the Syntax Diagrams" on page 427.

Command Parameters

A description of the parameters available to the command.

Usage Notes

Other information.

See Also

A cross-reference to related information.

db2admin - DB2 Administration Server

This utility is used to manage the DB2 Administration Server. For more information about the DB2 Administration Server, see the *Administration Guide*.

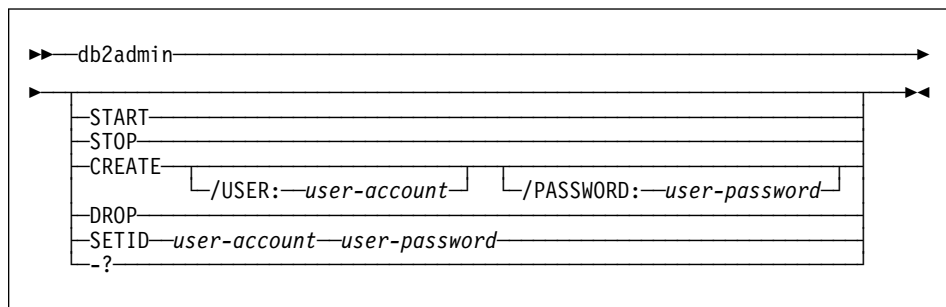
Authorization

Local administrator on Windows operating systems, or a system administrator on OS/2.

Required Connection

None

Command Syntax



Command Parameters

Note: If no parameters are specified, and the DB2 Administration Server exists, this command returns the DB2 Administration Server instance.

START

Start the DB2 Administration Server.

STOP

Stop the DB2 Administration Server.

CREATE

/USER: *user-account*

/PASSWORD: *user-password*

Create the DB2 Administration Server. If a user name and password are specified, the DB2 Administration Server instance will be associated with this user account. If the specified values are not valid, the utility returns an authentication error. The specified user account must be a valid SQL identifier, and must exist in the security database. It is recommended that a user account be specified to ensure that all DB2 Administration Server functions can be accessed.

DROP

Deletes the DB2 Administration Server instance.

db2admin - DB2 Administration Server

SETID *user-account/user-password*

Establishes or modifies the user account associated with the DB2 Administration Server instance.

-?

Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

db2adutl - Work with ADSM Archived Images

db2adutl - Work with ADSM Archived Images

Permits a user to query, extract, and delete backups, logs, and load copy images saved using ADSM. The utility is installed in the `INSTHOME/sql1ib/misc` directory on UNIX based systems, and in the `\sql1ib\misc` directory on OS/2 or the Windows operating system.

For a complete description of this command, see the *Administration Guide*.

db2audit - Audit Facility Administrator Tool

db2audit - Audit Facility Administrator Tool

DB2 provides an audit facility to assist in the detection of unknown or unanticipated access to data. The DB2 audit facility generates and permits the maintenance of an audit trail for a series of predefined database events. The records generated from this facility are kept in an audit log file. The analysis of these records can reveal usage patterns which would identify system misuse. Once identified, actions can be taken to reduce or eliminate such system misuse. The audit facility acts at an instance level, recording all instance level activities and database level activities.

Authorized users of the audit facility can control the following actions within the audit facility, using **db2audit**:

- Start recording auditable events within the DB2 instance.
- Stop recording auditable events within the DB2 instance.
- Configure the behavior of the audit facility.
- Select the categories of the auditable events to be recorded.
- Request a description of the current audit configuration.
- Flush any pending audit records from the instance and write them to the audit log.
- Extract audit records by formatting and copying them from the audit log to a flat file or ASCII delimited files. Extraction is done for one of two reasons: In preparation for analysis of log records, or in preparation for pruning of log records.
- Prune audit records from the current audit log.

For a complete description of this command, see the *Administration Guide*.

db2atld - Autoloader

Autoloader is a tool for splitting and loading data in an MPP environment. This utility can:

- Transfer data from one system (MVS, for example) to an AIX system (RS/6000 or SP2)
- Partition or split data in parallel
- Load data simultaneously on corresponding nodes.

For a complete description of this command, see the *Administration Guide*.

See Also

"db2split - Data Declustering Tool" on page 53
"LOAD" on page 276.

db2batch - Benchmark Tool

db2batch - Benchmark Tool

Reads SQL statements from either a flat file or standard input, dynamically prepares and describes the statements, and returns an answer set. This tool is provided in the misc subdirectory of the instance sql11ib directory.

Authorization

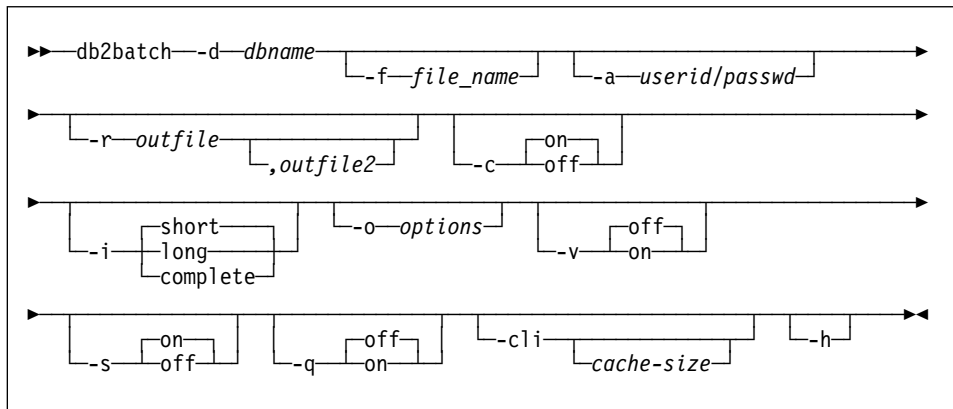
One of the following:

sysadm

Required Connection

None. This command establishes a database connection.

Command Syntax



Command Parameters

-d *dbname*

An alias name for the database against which SQL statements are to be applied. The default is the value of the **DB2DBDFT** environment variable.

-f *file_name*

Name of an input file containing SQL statements. The default is standard input.

Identify comment text with two hyphens at the start of each line, that is, `--<comment>`. If it is to be included in the output, mark the comment as follows: `--#COMMENT <comment>`.

A *block* is a number of SQL statements that are treated as one, that is, information is collected for all of those statements at once, instead of one at a time. Identify the beginning of a block of queries as follows: `--#BGBLK`. Identify the end of a block of queries as follows: `--#E0BLK`.

db2batch - Benchmark Tool

Specify one or more control options as follows: `--#SET <control option> <value>`. Valid control options are:

ROWS_FETCH

Number of rows to be fetched from the answer set. Valid values are -1 to n . The default value is -1 (all rows are to be fetched).

ROWS_OUT

Number of fetched rows to be sent to output. Valid values are -1 to n . The default value is -1 (all fetched rows are to be sent to output).

PERF_DETAIL

Specifies the level of performance information to be returned. Valid values are:

- 0* No timing is to be done.
- 1* Return elapsed time only.
- 2* Return elapsed time and CPU time.
- 3* Return a summary of monitoring information.
- 4* Return a snapshot for the database manager, the database, the application, and the statement (the latter is returned only if autocommit is off, and single statements, not blocks of statements, are being processed).
- 5* Return a snapshot for the database manager, the database, the application, and the statement (the latter is returned only if autocommit is off, and single statements, not blocks of statements, are being processed). Also return a snapshot for the bufferpools, table spaces and FCM (an FCM snapshot is only available in a multi-node environment).

The default value is 1 . A value >1 is only valid on DB2 Version 2 servers.

DELIMITER

A one- or two-character end-of-statement delimiter. The default value is a semicolon (;).

SLEEP

Number of seconds to sleep. Valid values are 1 to n .

PAUSE

Prompts the user to continue.

TIMESTAMP

Generates a time stamp.

-a *userid/passwd*

Name and password used to connect to the database. The slash (/) must be included.

-r *outfile*

An output file that will contain the query results. An optional *outfile2* will contain a results summary. The default is standard output.

-c

Automatically commit changes resulting from each SQL statement.

db2batch - Benchmark Tool

-i

An elapsed time interval (in seconds).

short

The time taken to open the cursor, complete the fetch, and close the cursor.

long

The elapsed time from the start of one query to the start of the next query, including pause and sleep times, and command overhead.

complete

The time to prepare, execute, and fetch, expressed separately.

-o options

Control options. Valid options are:

f *rows_fetch*

Number of rows to be fetched from the answer set. Valid values are -1 to *n*. The default value is -1 (all rows are to be fetched).

r *rows_out*

Number of fetched rows to be sent to output. Valid values are -1 to *n*. The default value is -1 (all fetched rows are to be sent to output).

p *perf_detail*

Specifies the level of performance information to be returned. Valid values are:

0 No timing is to be done.

1 Return elapsed time only.

2 Return elapsed time and CPU time.

3 Return a summary of monitoring information.

4 Return a snapshot for the database manager, the database, the application, and the statement (the latter is returned only if autocommit is off, and single statements, not blocks of statements, are being processed).

5 Return a snapshot for the database manager, the database, the application, and the statement (the latter is returned only if autocommit is off, and single statements, not blocks of statements, are being processed). Also return a snapshot for the bufferpools, table spaces and FCM (an FCM snapshot is only available in a multi-node environment).

o *query_optimization_class*

Sets the query optimization class. For a description of valid values, see the *Administration Guide*.

e *explain_mode*

Sets the explain mode under which **db2batch** runs. The explain tables must be created prior to using this option. Valid values are:

0 Run query only (default).

db2batch - Benchmark Tool

- 1 Populate explain tables only. This option populates the explain tables and causes explain snapshots to be taken.
- 2 Populate explain tables and run query. This option populates the explain tables and causes explain snapshots to be taken.

-v

Verbose. Send information to standard error during query processing. The default value is off.

-s

Summary Table. Provide a summary table for each query or block of queries, containing elapsed time (if selected), CPU times (if selected), the rows fetched, and the rows printed. The arithmetic and geometric means for elapsed time and CPU times are provided if they were collected.

-q

Query output. Valid values are:

on Print only the *non-delimited* output of the query.

off Print the output of the query and all associated information. This is the default.

del Print only the *delimited* output of the query.

-p

Parallel (MPP only). Valid values are:

-s Single table or collocated join query. If this option is specified, the NODENUMBER function will be added to the WHERE clause of the query, and a temporary table will not be created. This option is valid only if the query contains a single table in the FROM clause, or if the tables contained in the FROM clause are collocated.

-t table Specifies the table to use for an INSERT INTO statement. If the query contains multiple tables in the FROM clause, and the tables are not collocated, the result set must first be inserted into a temporary table, and then a SELECT from this temporary table must be performed on all nodes.

If neither the *-s* nor the *-t* option is specified, the tool creates a temporary table by default.

If a *local* output file is specified (using the *-r* option), the output from each node will go into a separate file with the same name on each node). If a file that is on an NFS-mounted file system is specified, all of the output will go into this file.

-cli

Run **db2batch** in CLI mode. The default is to use embedded dynamic SQL. The statement memory can be set manually, using the *cache-size* parameter.

cache-size

Size of the statement memory, expressed as number of statements. The default value is 25. If the utility encounters an SQL statement that has

db2batch - Benchmark Tool

- already been prepared, it will reuse the old plans. This parameter can only be set when **db2batch** is run in CLI mode.
- h** Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

Example

For a detailed discussion on the use of **db2batch**, see the *Administration Guide*.

Usage Notes

Although SQL statements can be up to 32 698 characters in length, no text line in the input file can exceed 3 898 characters, and long statements must be divided among several lines. Statements must be terminated by a delimiter (the default is a semicolon).

SQL statements are executed with the repeatable read (RR) isolation level.

See Also

“db2sql92 - SQL92 Compliant SQL Statement Processor” on page 54.

db2bfd - Bind File Description Tool

Displays the contents of a bind file. This utility, which can be used to examine and to verify the SQL statements within a bind file, as well as to display the precompile options used to create the bind file, may be helpful in problem determination related to an application's bind file.

On UNIX based systems, the tool is located in the `misc` subdirectory of the `sql11b` directory of the instance; on OS/2 or the Windows operating system, it is located in the `bin` subdirectory of the `sql11b` directory of the instance.

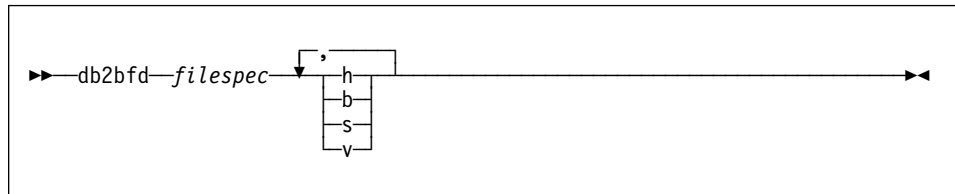
Authorization

None

Required Connection

None

Command Syntax



Command Parameters

filespec

Name of the bind file whose contents are to be displayed.

h

Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

b

Display the bind file header.

s

Display the SQL statements.

v

Display the host variable declarations.

db2cc - Start Control Center

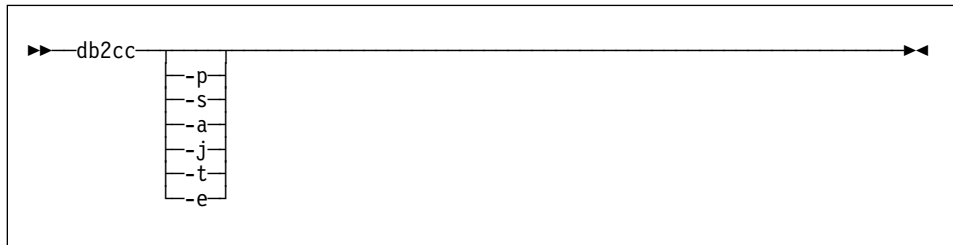
db2cc - Start Control Center

Starts the Control Center. The Control Center is an easy-to-use graphical interface that displays database objects (such as databases, tables, and packages) and their relationship to one another.

Authorization

sysadm

Command Syntax



Command Parameters

- p** Opens the Command Center.
- s** Opens the Script Center.
- a** Opens the Alert Center.
- j** Opens the Journal.
- t** Opens the Tools Settings notebook.
- e** Opens the Event Analyzer.

Usage Notes

Note: The following commands:

- “GET ADMIN CONFIGURATION” on page 171
- “RESET ADMIN CONFIGURATION” on page 352
- “UPDATE ADMIN CONFIGURATION” on page 405

cannot normally be issued from within the Command Center. This can only be done if the Command Center was started from a DB2 Administration Server instance. To execute these commands from any other instance, use the command line processor (CLP).

db2cc - Start Control Center

For general information about the Control Center, see the *Administration Getting Started* book. Detailed information is provided through the online help facility within the Control Center.

db2cidmg - Remote Database Migration

db2cidmg - Remote Database Migration

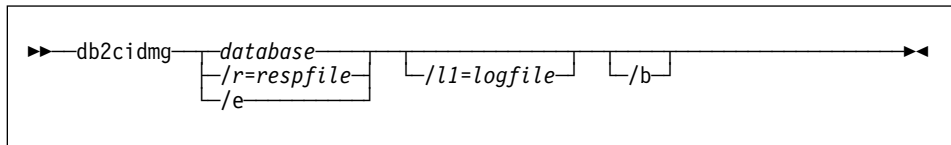
Supports remote unattended migration in the Configuration, Installation, and Distribution (CID) architecture environment.

Authorization

One of the following:

sysadm
dbadm

Command Syntax



Command Parameters

database

Specifies an alias name for the database which is to be migrated to DB2 Version 5. If not specified, a response file or */e* must be provided for program invocation. Note that the database alias must be cataloged on the target workstation. However, it can be a local or a remote database.

/r

Specifies a response file to be used for CID migration. The response file is an ASCII file containing a list of databases which are to be migrated. If not specified, a database alias or */e* must be provided for program invocation.

/e

Indicates that every single database cataloged in the system database directory is to be migrated. If */e* is not specified, a database alias or a response file must be provided.

/ll

Specifies the path name of the file to which error log information from remote workstations can be copied after the migration process is completed. If more than one database is specified in the response file, the log information for each database migration is appended to the end of the file. Regardless of whether */ll* is specified or not, a log file with the name DB2CIDMG.LOG is generated and kept in the workstation's file system where the database migration has been performed.

/b

Indicates that all packages in the database are to be rebound once migration is complete.

db2ckmig - Database Pre-migration Tool

db2ckmig - Database Pre-migration Tool

Verifies that a database can be migrated. For detailed information about using this tool, see one of the *Quick Beginnings* books.

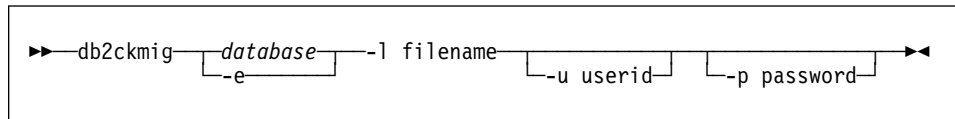
Authorization

sysadm

Required Connection

None

Command Syntax



Command Parameters

database

Specifies an alias name of a database to be scanned.

-e

Specifies that all local cataloged databases are to be scanned.

-l

Specifies a log file to keep a list of errors and warnings generated for the scanned database.

-u

Specifies the user ID of the system administrator.

-p

Specifies the password of the system administrator's user ID.

Usage Notes

To verify the state of a database:

1. Logon as the instance owner.
2. Issue the **db2ckmig** command.
3. Check the log file. If it shows errors, see one of the *Quick Beginnings* books for suggested corrective actions.

Note: The log file displays the errors that occur when the **db2ckmig** command is run. Check that the log is empty before continuing with the migration process.

db2cli - DB2 Interactive CLI

db2cli - DB2 Interactive CLI

Launches the interactive Call Level Interface environment for design and prototyping in CLI. Located in the `sqllib/samples/cli/` subdirectory of the home directory of the database instance owner.

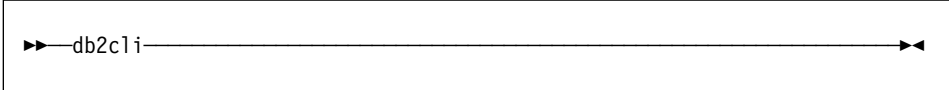
Authorization

None

Required Connection

None

Command Syntax



```
▶—db2cli—▶
```

Command Parameters

None

Usage Notes

DB2 Interactive CLI consists of a set of commands that can be used to design, prototype, and test CLI function calls. It is a programmers' testing tool provided for the convenience of those who want to use it, and IBM makes no guarantees about its performance. DB2 Interactive CLI is not intended for end users, and so does not have extensive error-checking capabilities.

Two types of commands are supported:

CLI commands

Commands that correspond to (and have the same name as) each of the function calls that is supported by IBM CLI

Support commands

Commands that do not have an equivalent CLI function.

Commands can be issued interactively, or from within a file. Similarly, command output can be displayed on the terminal, or written to a file. A useful feature of the CLI command driver is the ability to capture all commands that are entered during a session, and to write them to a file, thus creating a *command script* that can be rerun at a later time.

For more information about this utility, see the file `intcli.doc`, which is also located in the `sqllib/samples/cli/` subdirectory of the home directory of the database instance owner.

db2cmd - Open DB2 Command Window

db2cmd - Open DB2 Command Window

Opens the CLP-enabled DB2 window, and initializes the DB2 command line environment. Issuing this command is equivalent to clicking on the *DB2 Command Window* icon.

This command is available on Windows NT only.

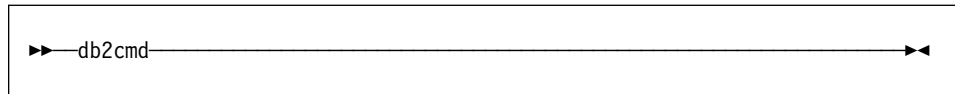
Authorization

None

Required Connection

None

Command Syntax



A diagram showing the command syntax for db2cmd. It consists of a horizontal line with a right-pointing arrowhead at the left end and a left-pointing arrowhead at the right end. The text "db2cmd" is positioned on the line, slightly to the right of the left arrowhead.

Command Parameters

None

db2drdat - DRDA Trace

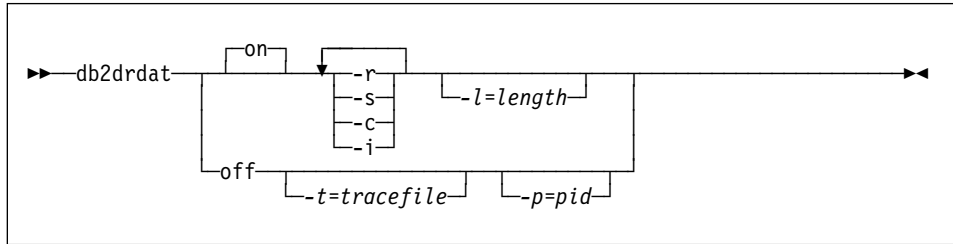
db2drdat - DRDA Trace

Allows the user to capture the DRDA data stream exchanged between a DRDA Application Requestor (AR) and the DB2 UDB DRDA Application Server (AS). Although this tool is most often used for problem determination, by determining how many sends and receives are required to execute an application, it can also be used for performance tuning in a client/server environment.

Authorization

None

Command Syntax



Command Parameters

- on* Turns on AS trace events (all if none specified).
- off* Turns off AS trace events.
- r* Traces DRDA requests received from the DRDA AR.
- s* Traces DRDA replies sent to the DRDA AR.
- c* Traces the SQLCA received from the DRDA server on the host system. This is a formatted, easy-to-read version of *not null* SQLCAs.
- i* Includes time stamps in the trace information.
- l* Specifies the size of the buffer used to store the trace information.
- p* Traces events only for this process. If *-p* is not specified, all agents with incoming DRDA connections on the server are traced.

Note: The *pid* to be traced can be found in the *agent* field returned by "LIST APPLICATIONS" on page 236.

db2drdat - DRDA Trace

-t

Specifies the destination for the trace. If a file name is specified without a complete path, missing information is taken from the current path.

Note: If *tracefile* is not specified, messages are directed to `db2drdat.dmp` in the current directory.

Usage Notes

Do not issue **db2trc** commands while **db2drdat** is active (for information about the **db2trc** command, see the *Troubleshooting Guide*).

db2drdat writes the following information to *tracefile*:

1. -r
 - Type of DRDA request
 - Receive buffer.
2. -s
 - Type of DRDA reply/object
 - Send buffer.
3. CPI-C error information
 - Severity
 - Protocol used
 - API used
 - Local LU name
 - Failed CPI-C function
 - CPI-C return code.

The command returns an exit code. A zero value indicates that the command completed successfully, and a nonzero value indicates that the command was not successful.

Note: If **db2drdat** sends the output to a file that already exists, the old file will be erased unless the permissions on the file do not allow it to be erased, in which case the operating system will return an error.

db2empfa - Enable Multi-page File Allocation

db2empfa - Enable Multi-page File Allocation

Enables multi-page file allocation for a database.

Scope

This command only affects the node on which it is executed.


Authorization

sysadm

Required Connection

None. This command establishes a database connection.

Command Syntax



► db2empfa database-alias ◄

Command Parameters

database-alias

Specifies the alias of the database for which multi-page file allocation is to be enabled.

Usage Notes

This utility:

- Connects to the database partition on a node (where applicable) in exclusive mode
- In all SMS table spaces, allocates empty pages to fill up the last extent in all data and index files which are larger than one extent
- Changes the value of the database configuration parameter *multipage_alloc* to YES
- Disconnects.

Since **db2empfa** connects to the database partition on a node in exclusive mode, it cannot be run concurrently on the catalog node, or on any other node.

db2eva - Event Analyzer

Starts the event analyzer, allowing the user to trace performance data produced by DB2 event monitors that have their data directed to files. See the *System Monitor Guide and Reference* for more information on event monitors.

Authorization

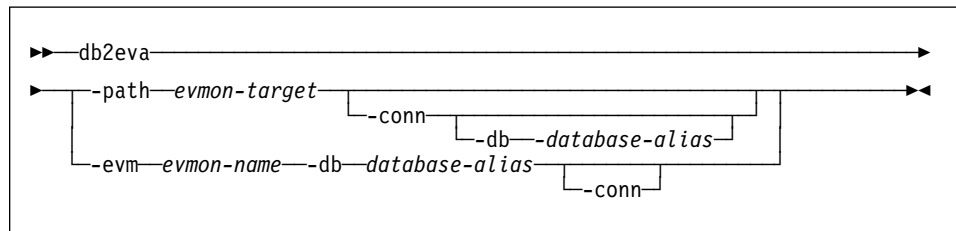
None, unless connecting to the database and selecting from the catalogs (-evm, -db, and -conn); then one of the following is required:

```
sysadm
sysctrl
sysmaint
dbadm
```

Required Connection

None

Command Syntax



Command Parameters

-path *evmon-target*

Specifies the directory containing the event monitor trace files.

-conn

Requests that **db2eva** maintain a connection to the database specified with -db, or if -db is not used, then to the database specified in the event monitor trace header. Maintaining a connection allows the event analyzer to obtain information not contained in the trace files (for example, the text for static SQL). A statement event record contains the package creator, package, and section number; when -conn is specified, **db2eva** can retrieve the text from the database system catalog (sysibm.sysstmt).

-db *database-alias*

Specifies the name of the database defined for the event monitor. If -path is specified, the database name in the event monitor trace header is overridden.

db2eva - Event Analyzer

Usage Notes

Although there is no required connection, **db2eva** will attempt to connect to the database if the `-conn`, or the `-evm` and the `-db` options are used. If the user can access the database and has the appropriate authorization, the SQL text for static statements can be displayed. Without the required access or authority, only the text for dynamic statements is available.

There are two methods for reading event monitor traces:

1. Specifying the directory where the trace files are located (using the `-path` option). This allows users to move trace files from a server and analyze them locally. This can be done even if the event monitor has been dropped.
2. Specifying the database and event monitor names allows automatic location of the trace files. The event analyzer connects to the database, and issues a `select target from sysibm.syseventmonitors` to locate the directory where the event monitor writes its trace files. The connection is then released, unless `-conn` was specified. This method cannot be used if the event monitor has been dropped.

Note: The event analyzer can be used to analyze the data produced by an active event monitor. However, event monitors buffer their data before writing it to disk; therefore, some information may be missing. Turn off the event monitor, thereby forcing it to flush its buffers.

db2evmon - Event Monitor Productivity Tool

db2evmon - Event Monitor Productivity Tool

Formats event monitor file and named pipe output, and writes it to standard output. Located in the `misc` subdirectory of the `sql11ib` directory of the instance.

Note: This productivity tool is provided *as is*, without any warranty of any kind, including the warranties of merchantability and fitness for a particular purpose, which are expressly disclaimed.

Authorization

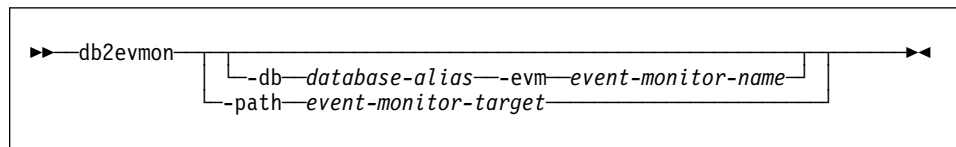
None, unless connecting to the database (`-evm`, `-db`); then, one of the following is required:

sysadm
sysctrl
sysmaint
dbadm

Required Connection

None

Command Syntax



Command Parameters

- db** *database-alias*
Specifies the database whose data is to be displayed.
- evm** *event-monitor-name*
The one-part name of the event monitor. An ordinary or delimited SQL identifier.
- path** *event-monitor-target*
Specifies the directory containing the event monitor trace files.

Usage Notes

If the data is being written to files, the tool formats the files for display using standard output. In this case, the monitor is turned on first, and any event data in the files is displayed by the tool. To view any data written to files after the tool has been run, reissue **db2evmon**.

If the data is being written to a pipe, the tool formats the output for display using standard output as events occur. In this case, the tool is started *before* the monitor is turned on.

db2exfmt - Explain Table Format Tool

db2exfmt - Explain Table Format Tool

Formats the contents of the explain tables. This tool is located in the `misc` subdirectory of the instance `sql11ib` directory.

For a complete description of this command, see the *Administration Guide*.

See Also

“db2expln - DB2 SQL Explain Tool” on page 27.

db2expln - DB2 SQL Explain Tool

Describes the access plan selection for static SQL statements in packages that are stored in the DB2 common server system catalogs. Given a database name, package name, package creator, and section number, the tool interprets and describes the information in these catalogs. This tool is located in the `misc` subdirectory of the instance `sqllib` directory.

For a complete description of this command, see the *Administration Guide*.

See Also

“db2exfmt - Explain Table Format Tool” on page 26.

db2flsn - Find Log Sequence Number

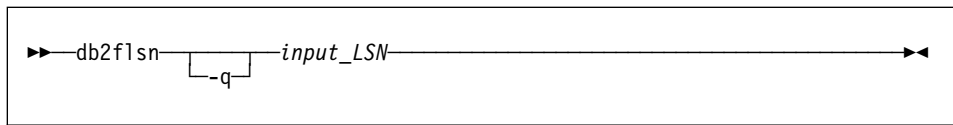
db2flsn - Find Log Sequence Number

Returns the name of the file that contains the log record identified by a specified log sequence number (LSN).

Authorization

None

Command Syntax



Command Parameters

-q

Specifies that only the log file name be printed. No error or warning messages will be printed, and status can only be determined through the return code. Valid error codes are:

- -100 Invalid input
- -101 Cannot open LFH file
- -102 Failed to read LFH file
- -103 Invalid LFH
- -104 Database is not recoverable
- -105 LSN too big
- -500 Logical error.

Other valid return codes are:

- 0 Successful execution
- 99 Warning: the result is based on the last known log file size.

input_LSN

A 12-byte string that represents the internal (6-byte) hexadecimal value with leading zeros.

db2flsn - Find Log Sequence Number

Examples

```
db2flsn 000000BF0030
    Given LSN is contained in log file S0000002.LOG
```

```
db2flsn -q 000000BF0030
    S0000002.LOG
```

```
db2flsn 000000BE0030
    Warning: the result is based on the last known log file size.
    The last known log file size is 23 4K pages starting from log extent 2.
```

```
    Given LSN is contained in log file S0000001.LOG
```

```
db2flsn -q 000000BE0030
    S0000001.LOG
```

Usage Notes

The log header control file `sqllogctl.lfh` must reside in the current directory. Since this file is located in the database directory, the tool can be run from the database directory, or the control file can be copied to the directory from which the tool will be run.

The tool uses the `logfilsiz` database configuration parameter. DB2 records the three most recent values for this parameter, and the first log file that is created with each `logfilsiz` value; this enables the tool to work correctly when `logfilsiz` changes. If the specified LSN predates the earliest recorded value of `logfilsiz`, the tool uses this value, and returns a warning. The tool can be used with database managers prior to UDB Version 5.2; in this case, the warning is returned even with a correct result (obtained if the value of `logfilsiz` remains unchanged).

This tool can only be used with recoverable databases. A database is recoverable if it is configured with `logretain` or `userexit` on.

See Also

“sqlurlog - Asynchronous Read Log” API in the *API Reference*.

db2gov - DB2 Governor

db2gov - DB2 Governor

Monitors and changes the behavior of applications that run against a database. By default, a daemon is started on every logical node, but the front-end utility can be used to start a single daemon at a specific node to monitor the activity against the database partition at that node.

Each governor daemon collects statistics about the applications running against a database. It then checks these statistics against the rules that were specified in the governor configuration file that applies to that specific database. The governor then acts according to these rules. For example, a rule may indicate that an application is using too much resource. In this situation, the governor may change the application's priority or force it off the database, according to instructions specified in the governor configuration file.

For a complete description of this command, see the *Administration Guide*.

See Also

“db2govlg - DB2 Governor Log Query” on page 31.

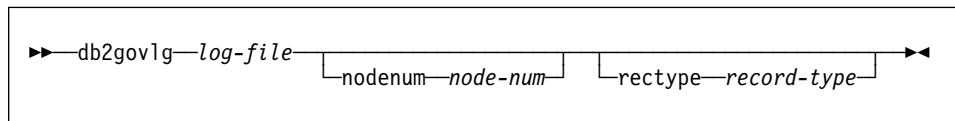
db2govlg - DB2 Governor Log Query

Extracts records of specified type from the governor log files (see “db2gov - DB2 Governor” on page 30). The DB2 governor monitors and changes the behavior of applications that run against a database.

Authorization

None

Command Syntax



Command Parameters

log-file

The base name of one or more log files that are to be queried.

nodenum *node-num*

Number of the node on which the governor is running.

rectype *record-type*

The type of record that is to be queried. Valid record types are:

- START
- FORCE
- NICE
- ERROR
- WARNING
- READCFG
- STOP
- ACCOUNT

See Also

“db2gov - DB2 Governor” on page 30.

db2icrt - Create Instance

db2icrt - Create Instance

Creates DB2 instances.

On UNIX based systems, this utility is located in the `DB2DIR/instance` directory, where `DB2DIR` represents `/usr/lpp/db2_05_00` on AIX, and `/opt/IBMDB2/V5.0` on both Solaris and HP-UX. On OS/2 or the Windows operating system, it is located in the `\sql1lib\bin` subdirectory.

For a complete description of this command, see one of the *Quick Beginnings* books.

db2idrop - Remove Instance

Removes a DB2 instance that was created by “db2icrt - Create Instance” on page 32. Removes the instance entry from the list of instances.

On UNIX based systems, this utility is located in the `DB2DIR/instance` directory, where `DB2DIR` represents `/usr/lpp/db2_05_00` on AIX, and `/opt/IBMDB2/V5.0` on both Solaris and HP-UX. On OS/2 or the Windows operating system, it is located in the `\sql11ib\bin` subdirectory.

For a complete description of this command, see one of the *Quick Beginnings* books.

db2ilist - List Instances

db2ilist - List Instances

Lists all the instances that are available on a system.

On UNIX based systems, this utility is located in the DB2DIR/bin directory, where DB2DIR represents /usr/lpp/db2_05_00 on AIX, and /opt/IBMDB2/V5.0 on both Solaris and HP-UX. On OS/2 or the Windows operating system, it is located in the \sql11ib\bin subdirectory.

For a complete description of this command, see one of the *Quick Beginnings* books.

db2imigr - Migrate Instance

Migrates an existing instance following installation of the database manager.

On UNIX based systems, this utility is located in the `DB2DIR/instance` directory, where `DB2DIR` represents `/usr/lpp/db2_05_00` on AIX, and `/opt/IBMd2/V5.0` on both Solaris and HP-UX. On OS/2 or the Windows operating system, it is located in the `\sql11ib\bin` subdirectory.

For a complete description of this command, see one of the *Quick Beginnings* books.

db2ipxad - Get IPX/SPX Internetwork Address

db2ipxad - Get IPX/SPX Internetwork Address

Returns the DB2 server's IPX/SPX internetwork address. This command *must* be issued locally from the DB2 server machine. Issuing the command from a remote client is not supported. The internetwork address can be used on a client machine to catalog an IPX/SPX node using "direct addressing". For more information, see one of the *Quick Beginnings* books.


Authorization

None

Required Connection

None

Command Syntax



```
▶—db2ipxad—▶
```

Command Parameters

None

See Also

"CATALOG IPX/SPX NODE" on page 126.

db2iupdt - Update Instances

Updates a specified DB2 instance to enable acquisition of a new system configuration or access to function associated with the installation or removal of certain product options.

On UNIX based systems, this utility is located in the `DB2DIR/instance` directory, where `DB2DIR` represents `/usr/lpp/db2_05_00` on AIX, and `/opt/IBMDB2/V5.0` on both Solaris and HP-UX. On OS/2 or the Windows operating system, it is located in the `\sql11ib\bin` subdirectory.

For a complete description of this command, see one of the *Quick Beginnings* books.

db2look - DB2 Statistics Extraction Tool

db2look - DB2 Statistics Extraction Tool

Generates the update statements required to make the catalog statistics of a test database match those of a production database.

It is often advantageous to have a test system contain a subset of the production system's data. However, access plans selected for such a test system are not necessarily the same as those that would be selected for the production system. Both the catalog statistics and the configuration parameters for the test system must be updated to match those of the production system. This tool queries the system catalogs of a database, and outputs table space, table, index, and column information about each table in that database.

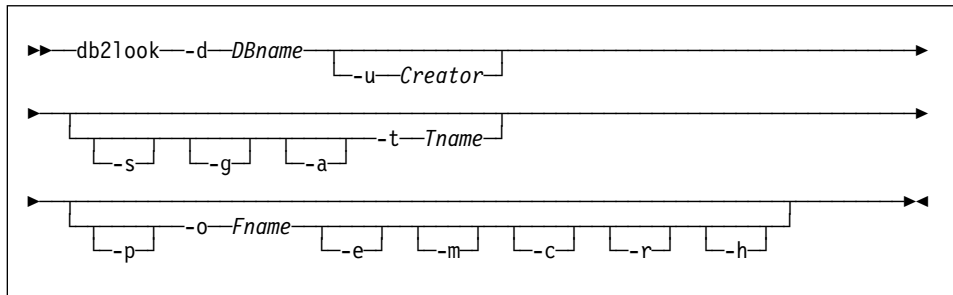
Authorization

SELECT privilege on the system catalogs.

Required Connection

None. This command establishes a database connection.

Command Syntax



Command Parameters

`-d DBname`

Alias name of the production database that is to be queried.

`-u Creator`

Creator ID. If option `-a` is specified, this parameter is ignored. If neither `-u` nor `-a` is specified, the environment variable **USER** is used.

`-s`

Generate a PostScript file.

Notes:

1. This option removes all LaTeX and `.tmp` PostScript files.
2. Required non-IBM software: LaTeX, dvips.
3. The `psfig.tex` file must be in the LaTeX input path.

db2look - DB2 Statistics Extraction Tool

- g** Use a graph to show fetch page pairs for indices.
- Notes:**
1. This option generates a *filename.ps* file, as well as the LaTeX file.
 2. Required non-IBM software: Gnuplot.
 3. The *psfig.tex* file must be in the LaTeX input path.
- a** Generate statistics for all creators. If specified, the *-u* parameter is ignored.
- t *Tname*** Table name. Limits the output to a particular table.
- p** Use plain text format.
- o *Fname*** If using LaTeX format, write the output to *filename.tex*. If using plain text format, write the output to *filename.txt*. If this option is not specified, output is written to standard output.
- e** Extract DDL statements to recreate database data objects. This option generates a CLP script containing DDL statements to recreate tables and indexes. The CLP script can then be run against another database to recreate the database. This option can be used in conjunction with the *-m* option. It does not currently support:
- Creation of triggers
 - Creation of UDFs
 - Reference to UDFs.
- m** Run the program in *mimic* mode. This option generates a CLP script containing all the UPDATE statements required to capture the catalog statistics of the production database. The CLP script can then be run against a smaller test database, and the updated test database can be used to validate access plans for production. The *-p*, *-g*, and *-s* options are ignored in *mimic* mode.
- c** Do not generate COMMIT statements in *mimic* mode. The default action is to generate COMMIT statements. In addition, do not generate CONNECT or CONNECT RESET statements. If option *-m* is not specified, this option is ignored.
- r** Do not include the RUNSTATS command in *mimic* mode. The default action is to issue the RUNSTATS command. If option *-m* is not specified, this option is ignored.
- h** Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

db2look - DB2 Statistics Extraction Tool

Examples

Write the statistics for tables created by USER in database DEPARTMENT in LaTeX format to file output.tex:

```
db2look -d department -o output
```

Write the statistics for all tables in database DEPARTMENT in LaTeX format to file output.tex:

```
db2look -a -d department -o output
```

Write the statistics for all tables in database DEPARTMENT in plain text format to file output.txt:

```
db2look -p -a -d department -o output
```

Write the statistics for all tables in database DEPARTMENT for user JAMES. Use a graph to show page fetch pairs. Save the LaTeX output in file output.tex. Save the PostScript output in file output.ps:

```
db2look -g -s -u james -d department -o output
```

Write the replica CLP commands for table EMPLOYEE in database PAYROLL created by anyone to file employee.mimic:

```
db2look -m -a -d payroll -t employee -o employee.mimic
```

db2migdr - Local Database Directory Migration

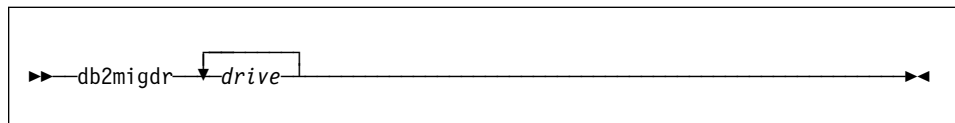
db2migdr - Local Database Directory Migration

Migrates a down-level local database directory to the current release format, so that databases stored in the local database directory can be accessed for migration.

Authorization

sysadm

Command Syntax



Command Parameters

drive

Specifies the drive where the local database directory can be found. The local database directory must be named `sqlbdir`, and it must be the first level subdirectory off the drive that is specified. The following example invokes the command to migrate the local database directory from drive `c` and from drive `f`:

```
db2migdr c f
```

db2mscs - Set up Windows NT Failover Utility

db2mscs - Set up Windows NT Failover Utility

Creates the infrastructure for DB2 to support failover on the Windows NT environment using MSCS support. This utility can be used to enable failover in both single-partition and partitioned database environments.

Run the utility once for each instance on its instance-owning machine. If there is only one DB2 instance running on one machine in the MSCS cluster, a hot-standby configuration is established. If an instance is running on each machine in the MSCS cluster, run the utility once on each instance-owning machine to set up a mutual takeover configuration.

This utility performs the following steps:

1. Reads the required MSCS and DB2 parameters from an input file called DB2MSCS.CFG. For information about the full set of input parameters, see the *Administration Guide*.
2. Validates the parameters in the input file.
3. Registers the DB2 resource type.
4. Creates the MSCS group (or groups) to contain the MSCS and DB2 resources.
5. Creates the IP resource.
6. Creates the Network Name resource.
7. Moves MSCS disks to the group.
8. Creates the DB2 resource (or resources).
9. Adds all required dependencies for the DB2 resource.
10. Converts the non-clustered DB2 instance into a clustered instance.
11. Brings all resources online.

For a complete description of this command, see the *Administration Guide*.

db2profc - DB2 SQLJ Profile Customizer

Processes an SQLJ profile containing embedded SQL statements. By default, a DB2 package is created in the database; this utility augments the profile with DB2-specific information for use at run time. This utility should be run after the SQLJ application has been translated, but before the application is run.

Authorization

One of the following:

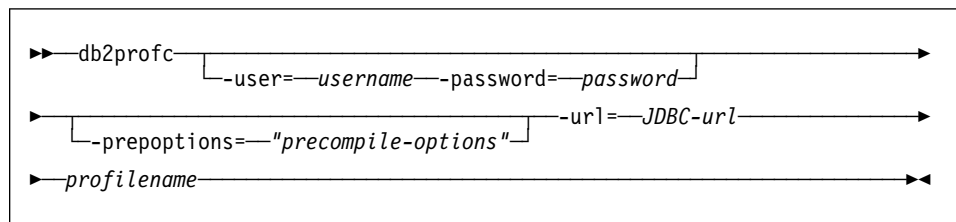
- *sysadm* or *dbadm* authority
- BINDADD privilege if a package does not exist, and one of:
 - IMPLICIT_SCHEMA authority on the database if the schema name of the package does not exist
 - CREATEIN privilege on the schema if the schema name of the package exists
- ALTERIN privilege on the schema if the package exists
- BIND privilege on the package if it exists.

The user also needs all privileges required to compile any static SQL statements in the application. Privileges granted to groups are not used for authorization checking of static statements. If the user has *sysadm* authority, but not explicit privileges to complete the bind, the database manager grants explicit *dbadm* authority automatically.

Required Connection

This command establishes a database connection.

Command Syntax



Command Parameters

-user= *username*

Specifies the name used when connecting to a database to perform profile customization.

-password= *password*

Specifies the password for the user name.

-preoptions= "*precompile-options*"

Specifies a list of precompile options to be used by the DB2 precompiler. For a list of supported precompile options, see the *Embedded SQL Programming Guide*.

db2profc - DB2 SQLJ Profile Customizer

The precompile option "PACKAGE USING package-name" specifies the name of the package that is to be generated by the precompiler. If a name is not entered, the name of the profile (minus extension and folded to uppercase) is used. Maximum length is 8 characters.

The precompile option "BINDFILE USING bind-file" specifies the name of the bind file that is to be generated by the precompiler. The file name must have an extension of .bnd. If a file name is not entered, the precompiler uses the name of the profile, and adds the .bnd extension. If a path is not provided, the bind file is created in the current directory.

-url= *JDBC-url*

Specifies a JDBC URL for establishing the database connection.

profilename

Specifies the name of a profile in which SQL statements are stored. When an SQLJ file is translated into a Java file, information about the SQL operations it contains is stored in SQLJ-generated resource files called profiles. Profiles are identified by the suffix *_SJProfileN* (where *N* is an integer) following the name of the original input file. They have a .ser extension. Profile names can be specified with or without the .ser extension.

Example

```
db2profc -user=username -password=password -url=JDBC-url
        -preoptions="bindfile using pgmname1.bnd package using pgmname1"
        pgmname_SJProfile1.ser
```

Usage Notes

For more information about SQLJ, see the *Embedded SQL Programming Guide*.

See Also

"db2profp - DB2 SQLJ Profile Printer" on page 45.

db2profp - DB2 SQLJ Profile Printer

Prints the contents of a DB2 customized version of a profile in plain text.

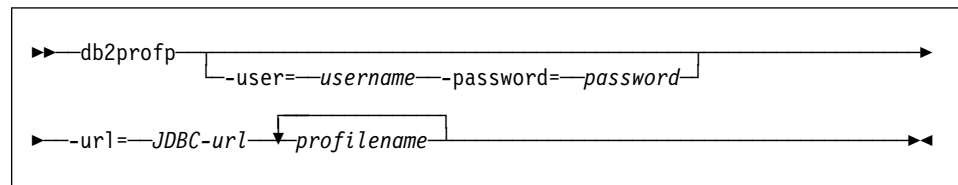
Authorization

None

Required Connection

This command establishes a database connection.

Command Syntax



Command Parameters

-user= *username*

Specifies the name used when connecting to a database to print the customized profile.

-password= *password*

Specifies the password for the user name.

-url= *JDBC-url*

Specifies a JDBC URL for establishing the database connection.

profilename

Specifies one or more profiles in which SQL statements are stored. When an SQLJ file is translated into a Java file, information about the SQL operations it contains is stored in SQLJ-generated resource files called profiles. Profiles are identified by the suffix *_SJProfileN* (where *N* is an integer) following the name of the original input file. They have a *.ser* extension. Profile names can be specified with or without the *.ser* extension.

Example

```
db2profp -user=username -password=password -url=JDBC-url  
pgmname_SJProfile1.ser
```

Usage Notes

For more information about SQLJ, see the *Embedded SQL Programming Guide*.

db2profp - DB2 SQLJ Profile Printer

| **See Also**

| "db2profc - DB2 SQLJ Profile Customizer" on page 43.

db2rbind - Rebind all Packages

Examines all packages in a database, and rebinds those packages which are marked as invalid, so that they are not rebound implicitly during application execution.

Authorization

One of the following:

sysadm
dbadm

Required Connection

None

Command Syntax

```
db2rbind database /l logfile [/u userid /p password]
```

Command Parameters

database

Specifies an alias name for the database whose packages are to be revalidated.

/l

Specifies the (optional) path and the (mandatory) file name to be used for recording errors that result from the package revalidation procedure.

/u

User ID (optional). This parameter must be specified if a password is specified.

/p

Password (optional). This parameter must be specified if a user ID is specified.

Usage Notes

This command uses the CLP REBIND command to attempt the revalidation of all packages in a database. Use of **db2rbind** is not mandatory. One may choose to allow package revalidation to occur implicitly when the package is first used, or to selectively revalidate particular packages with either the REBIND or the BIND command.

See Also

“BIND” on page 98
“REBIND” on page 332.

db2sampl - Create Sample Database

db2sampl - Create Sample Database

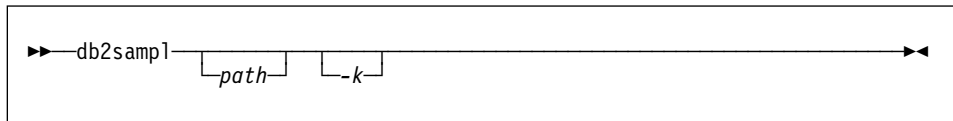
Creates a sample database named SAMPLE. For more information about this database, see the *SQL Reference*.

Authorization

One of the following:

sysadm
sysctrl

Command Syntax



Command Parameters

path

Specifies the path on which to create the SAMPLE database. The path is a single drive letter for OS/2 and Windows.

If a path is not specified, SAMPLE is created on the default database path (the *dftdbpath* parameter in the database manager configuration file). On UNIX based systems, the default is the HOME directory of the instance owner. On OS/2 or the Windows operating system, it is the root directory (where DB2 is installed).

-k

Creates primary keys on the following SAMPLE tables:

Table	Primary Key
DEPARTMENT	DEPTNO
EMPLOYEE	EMPNO
ORG	DEPTNUMB
PROJECT	PROJNO
STAFF	ID
STAFFG	ID (DBCS only)

Note: The path must be specified *before* this option.

Usage Notes

This command can only be executed from server nodes. SAMPLE cannot be created on nodes that are database clients only.

db2sampl - Create Sample Database

The SAMPLE database is created with the instance authentication type that is specified by the database manager configuration parameter *authentication* (see “GET DATABASE MANAGER CONFIGURATION” on page 188).

The qualifiers for the tables in SAMPLE are determined by the user ID issuing the command.

If SAMPLE already exists, **db2sampl** creates the tables for the user ID issuing the command, and grants the appropriate privileges.

db2set - DB2 Profile Registry Command

db2set - DB2 Profile Registry Command

Displays, sets, or removes DB2 profile variables. An external environment registry command that supports local and remote administration, via the DB2 Administration Server, of DB2's environment variables stored in the DB2 profile registry.

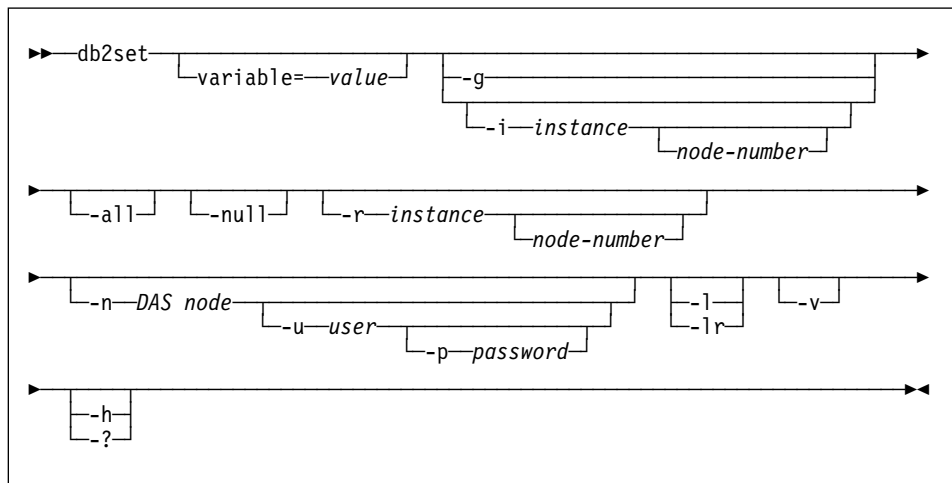
Authorization

sysadm

Required Connection

None

Command Syntax



Command Parameters

- g**
Access the global profile variables.
- i**
Specifies the instance profile to use instead of the current or default. The node number is a number listed in the `db2nodes.cfg` file.
- all**
Display all occurrences of the local environment variables as defined in:
 - The environment, denoted by [e]
 - The node level registry, denoted by [n]
 - The instance level registry, denoted by [i]
 - The global level registry, denoted by [g].

db2set - DB2 Profile Registry Command

-null	Set the value of the variable at the specified registry level to null. This avoids having to look up the value in the next registry level, as defined by the search order.
-r	Reset the profile registry for the given instance. The node number is a number listed in the <code>db2nodes.cfg</code> file.
-n	Specifies the remote DB2 administration server node name.
-u	Specifies the user ID to use for the administration server attachment.
-p	Specifies the password to use for the administration server attachment.
-l	List all instance profiles.
-lr	List all supported registry variables.
-v	Verbose mode.
-h/-?	Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

Examples

- Display all defined profiles (DB2 instances):
`db2set -l`
- Display all supported registry variables:
`db2set -lr`
- Display all defined global variables:
`db2set -g`
- Display all defined variables for the current instance:
`db2set`
- Display all defined values for the current instance:
`db2set -a11`
- Display all defined values for DB2COMM for the current instance:
`db2set -a11 DB2COMM`
- Reset all defined variables for the instance INST on node 3:
`db2set -r -i INST 3`
- Unset the variable DB2CHKPTR on the remote instance RMTINST through the DAS node RMTDAS using user ID MYID and password MYPASSWD:
`db2set -i RMTINST -n RMTDAS -u MYID -p MYPASSWD DB2CHKPTR=`

db2set - DB2 Profile Registry Command

- Set the variable DB2COMM to be TCPIP,IPXSPX,NETBIOS globally:
`db2set -g DB2COMM=TCPIP,IPXSPX,NETBIOS`
- Set the variable DB2COMM to be only TCPIP for instance MYINST:
`db2set -i MYINST DB2COMM=TCPIP`
- Set the variable DB2COMM to null at the given instance level:
`db2set -null DB2COMM`

Usage Notes

If no variable name is specified, the values of all defined variables are displayed. If a variable name *is* specified, only the value of that variable is displayed. To display all the defined values of a variable, specify *variable -all*. To display all the defined variables in all registries, specify *-all*.

To modify the value of a variable, specify *variable=*, followed by its new value. To set the value of a variable to NULL, specify *variable -null*.

Note: Changes to settings take effect after the instance has been restarted.

To delete a variable, specify *variable=*, followed by no value.

db2split - Data Declustering Tool

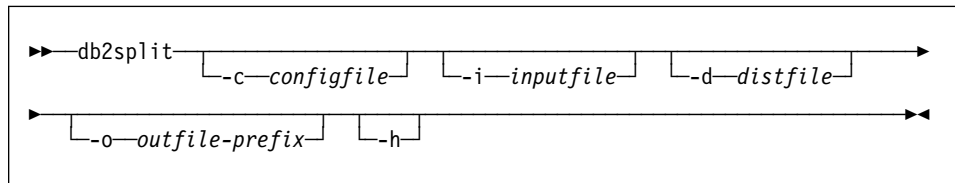
Partitions data across nodes in a multi-node environment.

For detailed information about **db2split**, see the *Administration Guide*.

Authorization

None

Command Syntax



Command Parameters

-c *configfile*

Name of the file containing configuration data required by the tool. The configuration file contains such information as the name of the input file, the position and length of the partitioning key, and the name of the log file. When using this option with a file name, the file name is assumed to contain all of the input information for the utility. The default value is `db2split.cfg`.

-i *inputfile*

Name of the file containing the data to be partitioned.

-d *distfile*

Name of the file containing an input partitioning map. If a customized partitioning map was created, it must be used.

-o *outfile-prefix*

The prefix of the output file. The utility appends a 5-character suffix (00000..00999) to the end of the prefix to generate the output file name.

-h

Display help information.

Usage Notes

The **-i**, **-d**, and **-o** options can be used to override the file names specified in the configuration file.

db2sql92 - SQL92 Compliant SQL Statement Processor

db2sql92 - SQL92 Compliant SQL Statement Processor

Reads SQL statements from either a flat file or standard input, dynamically describes and prepares the statements, and returns an answer set. Supports concurrent connections to multiple databases. This tool is provided in the `misc` subdirectory of the instance `sqllib` directory.

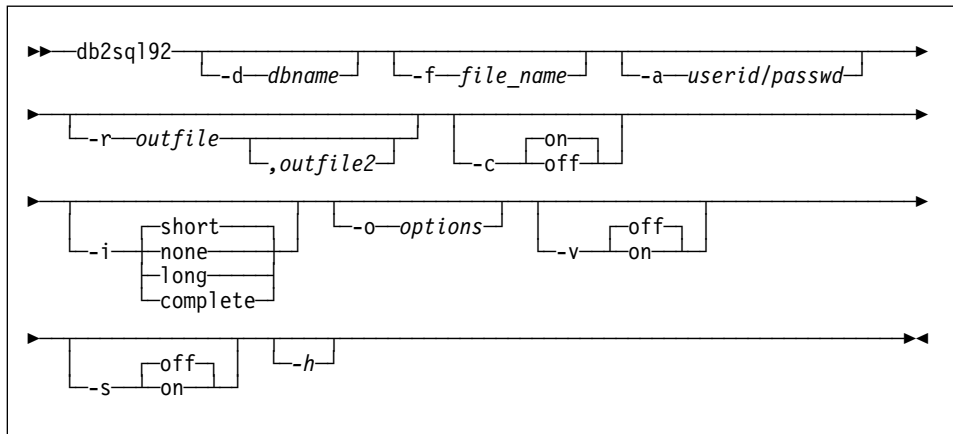
Authorization

`sysadm`

Required Connection

None. This command establishes a database connection.

Command Syntax



Command Parameters

-d *dbname*

An alias name for the database against which SQL statements are to be applied. The default is the value of the **DB2DBDFT** environment variable.

-f *file_name*

Name of an input file containing SQL statements. The default is standard input.

Identify comment text with two hyphens at the start of each line, that is, `--<comment>`. If it is to be included in the output, mark the comment as follows: `--#COMMENT <comment>`.

A *block* is a number of SQL statements that are treated as one, that is, information is collected for all of those statements at once, instead of one at a time. Identify the beginning of a block of queries as follows: `--#BGBLK`. Identify the end of a block of queries as follows: `--#E0BLK`.

db2sql92 - SQL92 Compliant SQL Statement Processor

Specify one or more control options as follows: `--#SET <control option> <value>`. Valid control options are:

ROWS_FETCH

Number of rows to be fetched from the answer set. Valid values are -1 to *n*. The default value is -1 (all rows are to be fetched).

ROWS_OUT

Number of fetched rows to be sent to output. Valid values are -1 to *n*. The default value is -1 (all fetched rows are to be sent to output).

AUTOCOMMIT

Specifies autocommit on or off. Valid values are ON or OFF. The default value is ON.

PAUSE

Prompts the user to continue.

TIMESTAMP

Generates a time stamp.

-a *userid/passwd*

Name and password used to connect to the database.

-r *outfile*

An output file that will contain the query results. An optional *outfile2* will contain a results summary. The default is standard output.

-c

Automatically commit changes resulting from each SQL statement.

-i

An elapsed time interval (in seconds).

none

Specifies that time information is not to be collected.

short

The run time for a query.

long

Elapsed time at the start of the next query.

complete

The time to prepare, execute, and fetch, expressed separately.

-o *options*

Control options. Valid options are:

f *rows_fetch*

Number of rows to be fetched from the answer set. Valid values are -1 to *n*. The default value is -1 (all rows are to be fetched).

r *rows_out*

Number of fetched rows to be sent to output. Valid values are -1 to *n*. The default value is -1 (all fetched rows are to be sent to output).

-v

Verbose. Send information to standard error during query processing. The default value is off.

db2sql92 - SQL92 Compliant SQL Statement Processor

- s** Summary Table. Provide a summary of elapsed times and CPU times, containing both the arithmetic and the geometric means of all collected values.
- h** Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

Usage Notes

The following can be executed from the **db2sql92** command prompt:

- All control options
- SQL statements
- CONNECT statements
- commit work
- help
- quit

This tool supports switching between different databases during a single execution of the program. To do this, issue a **CONNECT RESET** and then one of the following on the **db2sql92** command prompt (stdin):

```
connect to database  
connect to database USER userid USING passwd
```

SQL statements can be up to 32700 characters in length. Statements must be terminated by a semicolon.

SQL statements are executed with the repeatable read (RR) isolation level.

See Also

“db2batch - Benchmark Tool” on page 8.

db2start - Start DB2

Starts the current database manager instance background processes on a single node or on all the nodes defined in a multi-node environment. Start DB2 at the server before connecting to a database, precompiling an application, or binding a package to a database.

db2start can be executed as a system command or a CLP command. For a complete description of this command, see "START DATABASE MANAGER" on page 390.

db2stop - Stop DB2

db2stop - Stop DB2

Stops the current database manager instance.

db2stop can be executed as a system command or a CLP command. For a complete description of this command, see “STOP DATABASE MANAGER” on page 395.

db2tbst - Get Tablespace State

Accepts a hexadecimal table space state value, and returns the state. The state value is part of the output from “LIST TABLESPACES” on page 271.

Authorization

None

Required Connection

None

Command Syntax

```
▶▶—db2tbst—tablespace-state—————▶▶
```

Command Parameters

tablespace-state

A hexadecimal table space state value.

Example

The request `db2tbst 0x0000` produces the following output:

```
State = Normal
```

See Also

“LIST TABLESPACES” on page 271.

db2trc - Trace

db2trc - Trace

Activates DB2 trace facility tracing. DB2 uses its trace facility to trace events, dump trace data to a file, and format trace data into a readable form. Only use the trace facility when directed by DB2 Customer Service or by a technical support representative.

Authorization

On UNIX based systems, one of the following:

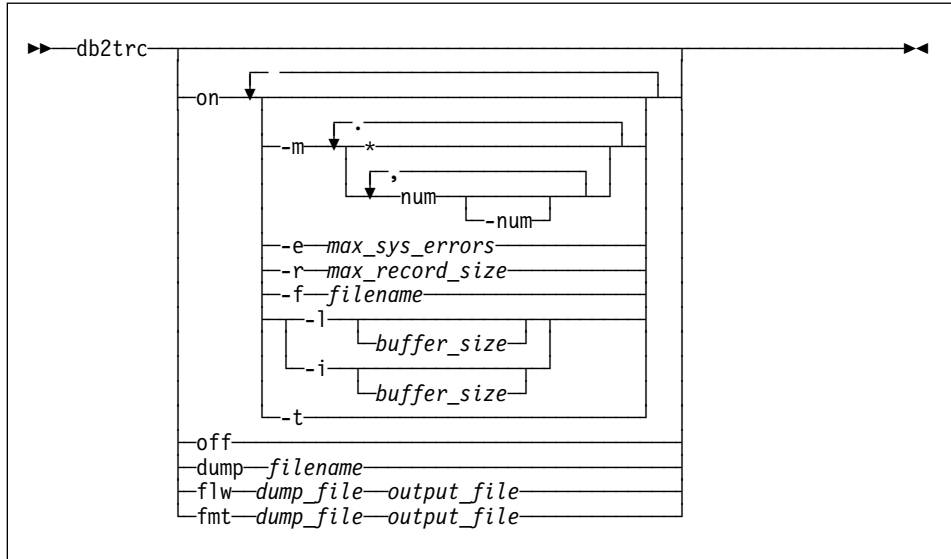
```
sysadm  
sysctrl  
sysmaint
```

On OS/2 or the Windows operating system, no authorization is required.

Required Connection

None

Command Syntax



Command Parameters

on

Use this parameter to start the DB2 trace facility. See the next section for information about the options for this parameter.

dump

After reproducing the error, dump the trace information to a file. The following command will put the information in the current directory in a file called db2trc.dmp:

```
db2trc dump db2trc.dmp
```

Specify a file name with this parameter. The file is saved in the current directory unless the path is explicitly specified.

off

After the trace is dumped to a file, turn the trace off by typing:

```
db2trc off
```

flw | fmt

After the trace is dumped to a binary file, confirm that it is taken by formatting it into an ASCII file. Use either the flw option (to sort by process or thread), or the fmt option (to list every event chronologically). For either option, specify the name of the dump file and the name of the output file that will be generated. For example:

```
db2trc flw db2trc.dmp db2trc.flw
```

Starting a DB2 Trace

To use the trace facility, first turn it on using **db2trc on**. Unless instructed otherwise by DB2 Customer Service, use the command without options. The default trace option values are:

- *mask* = * (trace everything)
- *max_sys_errors* = -1 (collect all)
- *max_record_size* = 16 KB
- Trace destination = -s (shared memory)
- Records to retain = -l (last)
- *buffer_size* = 64 KB if the trace destination is shared memory, or 1MB if the trace destination is a file.

If instructed to specify options to tailor the trace, use the following options as directed by DB2 Customer Service:

- m *mask* Specifies trace record types to focus the search. The *mask* variable consists of four one-byte masks separated by periods. The byte-masks act as a filter to accept or reject the trace record sent by DB2 for each event based on its ID. The four one-byte masks correspond to products, event types, components, and functions, respectively.
- e *max_sys_errors* Limits the number of DB2 internal system errors the trace will hold to *max_sys_errors*.
- r *max_record_size* Limits the size of trace records to *max_record_size* bytes. Longer trace records are truncated.
- f *filename* Specifies trace buffer location in shared memory or a file.

db2trc - Trace

- l [*buffer_size*] | -i [*buffer_size*]
'-l' ("l" as in "lucky") specifies that the last trace records are retained (that is, the first records are overwritten when the buffer is full). '-i' specifies that the initial trace records are retained (that is, no more records are written to the buffer once it is full). Use either of these options to specify the buffer size.
- t
Includes time stamps. Applicable to UNIX environments only, where the logging of time stamps affects performance.

Examples

For additional information about **db2trc**, including trace examples, see the *Troubleshooting Guide*.

Usage Notes

The **db2trc** command must be issued several times to turn tracing on, produce a dump file, format the dump file, and turn tracing off again. The parameter list shows the order in which the parameters should be used.

db2uiddl - Prepare Unique Index Conversion to V5 Semantics

db2uiddl - Prepare Unique Index Conversion to V5 Semantics

Facilitates the management of a staged migration of unique indexes on a user's own schedule. Generates CREATE UNIQUE INDEX statements for unique indexes on user tables. For detailed information about using this tool, see one of the *Quick Beginnings* books.

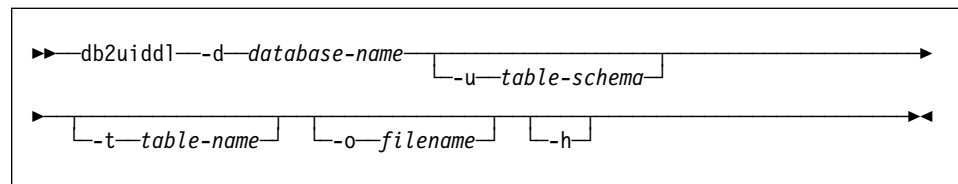
Authorization

sysadm

Required Connection

Database. This command automatically establishes a connection to the specified database.

Command Syntax



Command Parameters

-d *database-name*

The name of the database to be queried.

-u *table-schema*

Specifies the schema (creator user ID) of the tables that are to be processed. The default action is to process tables created by all user IDs.

-t *table-name*

The name of a table that is to be processed. The default action is to process all tables.

-o *filename*

The name of a file to which output is to be written. The default action is to write output to standard output.

-h

Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

Usage Notes

This tool can only be used on a database that has been migrated to DB2 Version 5.

Note: This tool was not designed to handle certain types of names. If a specific table name or table schema is a delimited identifier containing lower case characters, special characters, or blanks, it is preferable to request processing of *all* tables or schemas. The resulting output can be edited.

db2untag - Release Container Tag

db2untag - Release Container Tag

Removes the DB2 tag on a table space container. The tag is used to prevent DB2 from reusing a container in more than one table space. Displays information about the container tag, identifying the database with which the container is associated. Useful when it is necessary to release a container last used by a database that has since been deleted. If the tag is left behind, DB2 is prevented from using the resource in future.

Attention: This tool should only be used by informed system administrators.

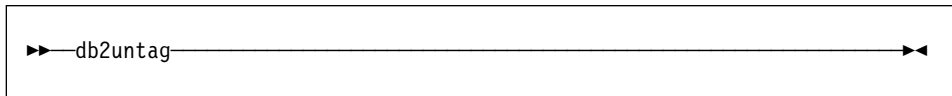
Authorization

The user needs read/write access to the container for a table space that is owned by the ID that created the database.

Required Connection

None

Command Syntax



Command Parameters

None

Usage Notes

An SQLCODE -294 (Container in Use error) is sometimes returned from create database or from create or alter table space operations, usually indicating a specification error on the operating system resource name when the container is already in use by another table space. A container can be used by only one table space at a time.

A system or database administrator who finds that the database which last used the container has been deleted, can use the **db2untag** tool if the container's tag was not removed. If the container is to be released, do one of the following:

- For SMS containers, remove the directory and its contents using the appropriate delete commands.
- For DMS containers, either delete the file or device, or let **db2untag** remove the container tag. The tool will leave such a DMS container otherwise unmodified.

See Also

“CREATE DATABASE” on page 143.

db2upd52 - Update Catalog to Support Version 5.2

db2upd52 - Update Catalog to Support Version 5.2

Updates catalog table data that is required to properly support new features in DB2 Version 5.2. These include support for the BIGINT data type, and associated functions in the SYSFUN schema.

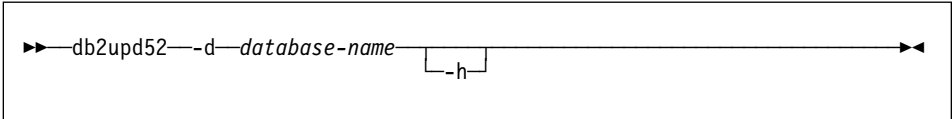
Authorization

sysadm

Required Connection

Database. This command automatically establishes a connection to the specified database.

Command Syntax



The diagram shows the command syntax for db2upd52. It starts with a right-pointing arrow followed by the command name 'db2upd52'. This is followed by an option '-d' and a placeholder 'database-name'. Below 'database-name' is a bracketed box containing '-h'. A long horizontal line with arrows at both ends extends to the right from the end of the 'database-name' placeholder.

Command Parameters

-d *database-name*

Specifies the database to which the catalog update is to be applied.

-h

Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

Usage Notes

This command can only be used if the server has been upgraded to DB2 Universal Database Version 5.2. The command should be issued once for each database defined under a DB2 instance that has been upgraded.

db2vexp - Dynamic Visual Explain

Explains a dynamic SQL statement, producing an access plan graph. Creates explain tables if they do not exist.

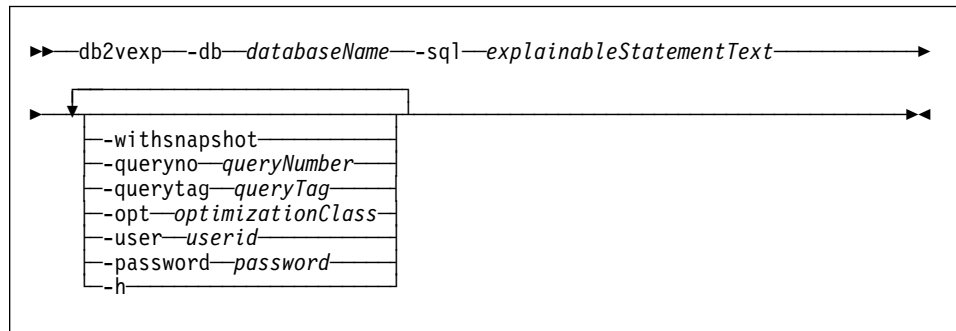
Authorization

The authorization rules are those defined for the SQL statement specified in the **db2vexp** command. For example, if a DELETE statement was used as the *explainableStatementText*, then the authorization rules for a DELETE statement apply. The current authorization ID must have INSERT privilege on the explain tables. If explain tables do not exist, the current authorization ID must also have CREATETAB privilege on the database. The explain tables will be created automatically.

Required Connection

None

Command Syntax



Command Parameters

- db** *databaseName*
The name of the database where a connection will be made.
- sql** *explainableStatementText*
The statement text to be dynamically explained, enclosed by double quotation marks.
- withsnapshot**
Generates more explain details.
- queryno** *queryNumber*
A query number to be associated with the dynamic explain.
- querytag** *queryTag*
A query tag to be associated with the dynamic explain. It can be a maximum of twenty characters enclosed by double quotation marks.
- opt** *optimizationClass*
Optimization class. Valid values are 0, 1, 2, 3, 5, 7, and 9. See SET CURRENT QUERY OPTIMIZATION in the *SQL Reference*.

db2vexp - Dynamic Visual Explain

- user** *userid*
Specifies the user ID used to connect to the database.
- password** *password*
Specifies the password used to connect to the database.
- h**
Displays help information. When this option is specified, all other options are ignored and only the help is displayed. On Windows platforms, */?* can also be used to specify the help option.

Usage Notes

A slash (/) can be used instead of a hyphen (-) to prefix a keyword.

Chapter 2. Command Line Processor (CLP)

This chapter explains how to invoke and use the command line processor, and describes CLP options.

Command Line Processor Invocation and Options

The **db2** command starts the command line processor. The CLP is used to execute database utilities, SQL statements and online help. It offers a variety of command options, and can be started in:

- Interactive input mode, characterized by the **db2 =>** input prompt
- Command mode, where each command must be prefixed by **db2**
- Batch mode, which uses the **-f** file input option.

Note: On Windows NT, “db2cmd - Open DB2 Command Window” on page 19 opens the CLP-enabled DB2 window, and initializes the DB2 command line environment. Issuing this command is equivalent to clicking on the *DB2 Command Window* icon.

“QUIT” on page 331 stops the command line processor. “TERMINATE” on page 398 also stops the command line processor, but removes the associated back-end process and frees any memory that is being used. TERMINATE is recommended if the database has been stopped, or if database configuration parameters have been changed.

Note: Existing connections should be reset before terminating the CLP.

The shell command (!), allows operating system commands to be executed from the interactive or the batch mode on UNIX based systems, and on OS/2 or the Windows operating system (!!s on UNIX, and !dir on OS/2 or the Windows operating system, for example).

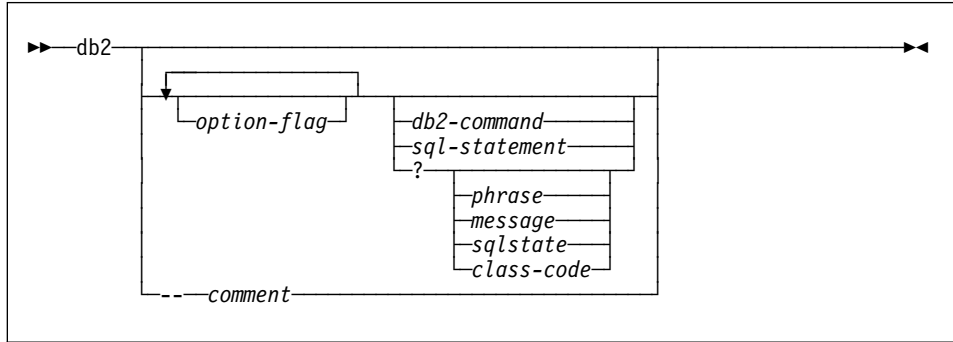
Note: Shell command support is not available on Windows 3.1.

Authorization

None

Command Line Processor Invocation and Options

Command Syntax



Command Parameters

option-flag

For a summary of valid CLP option flags, see Table 1 on page 71.

db2-command

Specifies a DB2 command. For a description of DB2 commands, see Chapter 3, "CLP Commands" on page 83.

sql-statement

Specifies an SQL statement.

?

Requests CLP general help.

? phrase

Requests the help text associated with a specified command or topic. If the database manager cannot find the requested information, it displays the general help screen.

? options requests a description and the current settings of the CLP options. ? help requests information about reading the online help syntax diagrams.

? message

Requests help for a message specified by a valid SQLCODE (? sql10007n, for example).

? sqlstate

Requests help for a message specified by a valid SQLSTATE.

? class-code

Requests help for a message specified by a valid class-code.

-- comment

Input that begins with the comment characters -- is treated as a comment by the command line processor.

Note: In each case, a blank space must separate the question mark (?) from the variable name.

Command Line Processor Invocation and Options

Options

The CLP command options can be specified by setting the command line processor **DB2OPTIONS** environment variable (which must be in uppercase), or with command line flags.

Users can set options for an entire session using **DB2OPTIONS**.

View the current settings for the option flags and the value of **DB2OPTIONS** using “LIST COMMAND OPTIONS” on page 238. Change an option setting from the interactive input mode or a command file using “UPDATE COMMAND OPTIONS” on page 409.

The command line processor sets options in the following order:

1. Sets up default options.
2. Reads **DB2OPTIONS** to override the defaults.
3. Reads the command line to override **DB2OPTIONS**.
4. Accepts input from UPDATE COMMAND OPTIONS as a final interactive override.

Table 1 summarizes the CLP option flags. These options can be specified in almost any sequence and combination. To turn an option on, prefix the corresponding option letter with a minus sign (-). To turn an option off, either prefix the option letter with a minus sign and follow the option letter with another minus sign, or prefix the option letter with a plus sign (+). For example, -c turns the auto-commit option on, and either -c- or +c turns it off. These option letters are not case sensitive, that is, -a and -A are equivalent.

Table 1 (Page 1 of 2). CLP Command Options

Option Flag	Description	Default Setting
-a	This option tells the command line processor to display SQLCA data.	OFF
-c	This option tells the command line processor to automatically commit SQL statements.	ON
-e{c s}	This option tells the command line processor to display SQLCODE or SQLSTATE. These options are mutually exclusive.	OFF
-filename	This option tells the command line processor to read command input from a file instead of from standard input.	OFF
-ifilename	This option tells the command line processor to log commands in a history file.	OFF
-o	This option tells the command line processor to display output data and messages to standard output.	ON
-p	This option tells the command line processor to display a command line processor prompt when in interactive input mode.	ON

Command Line Processor Invocation and Options

Option Flag	Description	Default Setting
-rfilename	This option tells the command line processor to write the report generated by a command to a file.	OFF
-s	This option tells the command line processor to stop execution if errors occur while executing commands in a batch file or in interactive mode.	OFF
-t	This option tells the command line processor to use a semicolon (;) as the statement termination character.	OFF
-tdx	This option tells the command line processor to define and to use x as the statement termination character.	OFF
-v	This option tells the command line processor to echo command text to standard output.	OFF
-w	This option tells the command line processor to display SQL statement warning messages.	ON
-zfilename	This option tells the command line processor to redirect all output to a file. It is similar to the -r option, but includes any messages or error codes with the output.	OFF

Example

The AIX command:

```
export DB2OPTIONS='+a -c +ec -o -p'
```

sets the following default settings for the session:

```
Display SQLCA - off
Auto Commit - on
Display SQLCODE - off
Display Output - on
Display Prompt - on
```

The following is a detailed description of these options:

Show SQLCA Data Option (-a):

Displays SQLCA data to standard output after executing a DB2 command or an SQL statement. The SQLCA data is displayed instead of an error or success message.

The default setting for this command option is OFF (+a or -a-).

The -o and the -r options affect the -a option; see the option descriptions for details.

Auto-commit Option (-c):

This option specifies whether each command or statement is to be treated independently. If set ON (-c), each command or statement is automatically

Command Line Processor Invocation and Options

committed or rolled back. If the command or statement is successful, it and all successful commands and statements that were issued before it with autocommit OFF (+c or -c-) are committed. If, however, the command or statement fails, it and all successful commands and statements that were issued before it with autocommit OFF are rolled back. If set OFF (+c or -c-), COMMIT or ROLLBACK must be issued explicitly, or one of these actions will occur when the next command with autocommit ON (-c) is issued.

The default setting for this command option is ON.

The auto-commit option does not affect any other command line processor option.

Example: Consider the following scenario:

1. db2 create database test
2. db2 connect to test
3. db2 +c "create table a (c1 int)"
4. db2 select c2 from a

The SQL statement in step 4 fails because there is no column named C2 in table A. Since that statement was issued with auto-commit ON (default), it rolls back not only the statement in step 4, but also the one in step 3, because the latter was issued with auto-commit OFF. The command:

```
db2 list tables
```

then returns an empty list.

Display SQLCODE/SQLSTATE Option (-e):

The -e{c|s} option tells the command line processor to display the SQLCODE (-ec) or the SQLSTATE (-es) to standard output. Options -ec and -es are not valid in CLP interactive mode.

The default setting for this command option is OFF (+e or -e-).

The -o and the -r options affect the -e option; see the option descriptions for details.

The display SQLCODE/SQLSTATE option does not affect any other command line processor option.

Example: To retrieve SQLCODE from the command line processor running on AIX, enter:

```
sqlcode='db2 -ec +o db2-command'
```

Read from Input File Option (-f):

The -f*filename* option tells the command line processor to read input from a specified file, instead of from standard input. *Filename* is an absolute or relative file name which may include the directory path to the file. If the directory path is not specified, the current directory is used.

When other options are combined with option -f, option -f must be specified last. For example:

```
db2 -tvf filename
```

Command Line Processor Invocation and Options

Note: This option cannot be changed from within the interactive mode.

The default setting for this command option is OFF (+f or -f-).

Commands are processed until “QUIT” on page 331 or “TERMINATE” on page 398 is issued, or an end-of-file is encountered.

If both this option and a database command are specified, the command line processor does not process any commands, and an error message is returned.

Input file lines which begin with the comment characters -- are treated as comments by the command line processor. Comment characters must be the first non-blank characters on a line.

If the -f*filename* option is specified, the -p option is ignored.

The read from input file option does not affect any other command line processor option.

Log Commands in History File Option (-l):

The -l*filename* option tells the command line processor to log commands to a specified file. This history file contains records of the commands executed and their completion status. *Filename* is an absolute or relative file name which may include the directory path to the file. If the directory path is not specified, the current directory is used. If the specified file or default file already exists, the new log entry is appended to that file.

When other options are combined with option -l, option -l must be specified last. For example:

```
db2 -tv1 filename
```

The default setting for this command option is OFF (+l or -l-).

The log commands in history file option does not affect any other command line processor option.

Display Output Option (-o):

The -o option tells the command line processor to send output data and messages to standard output.

The default setting for this command option is ON.

The interactive mode start-up information is not affected by this option. Output data consists of report output from the execution of the user-specified command, and SQLCA data (if requested).

The following options may be affected by the +o option:

- -r*filename*: Interactive mode start-up information is not saved.
- -e: SQLCODE or SQLSTATE is displayed on standard output even if +o is specified.
- -a: No effect if +o is specified. If -a, +o and -r*filename* are specified, SQLCA information is written to a file.

Command Line Processor Invocation and Options

If both `-o` and `-e` options are specified, the data and either the `SQLCODE` or the `SQLSTATE` are displayed on the screen.

If both `-o` and `-v` options are specified, the data is displayed, and the text of each command issued is echoed to the screen.

The display output option does not affect any other command line processor option.

Display DB2 Interactive Prompt Option (-p):

The `-p` option tells the command line processor to display the command line processor prompt when the user is in interactive mode.

The default setting for this command option is ON.

Turning the prompt off is useful when commands are being piped to the command line processor. For example, a file containing CLP commands could be executed by issuing:

```
db2 +p < myfile.clp
```

The `-p` option is ignored if the `-filename` option is specified.

The display DB2 interactive prompt option does not affect any other command line processor option.

Save to Report File Option (-r):

The `-filename` option causes any output data generated by a command to be written to a specified file, and is useful for capturing a report that would otherwise scroll off the screen. Messages or error codes are not written to the file. *Filename* is an absolute or relative file name which may include the directory path to the file. If the directory path is not specified, the current directory is used. New report entries are appended to the file.

The default setting for this command option is OFF (`+r` or `-r-`).

If the `-a` option is specified, `SQLCA` data is written to the file.

The `-r` option does not affect the `-e` option. If the `-e` option is specified, `SQLCODE` or `SQLSTATE` is written to standard output, not to a file.

If `-filename` is set in **DB2OPTIONS**, the user can set the `+r` (or `-r-`) option from the command line to prevent output data for a particular command invocation from being written to the file.

The save to report file option does not affect any other command line processor option.

Stop Execution on Command Error Option (-s):

When commands are issued in interactive mode, or from an input file, and syntax or command errors occur, the `-s` option causes the command line processor to stop execution and to write error messages to standard output.

The default setting for this command option is OFF (`+s` or `-s-`). This setting causes the command line processor to display error messages, continue execution of the remaining commands, and to stop execution only if a system error occurs (return code 8).

Command Line Processor Invocation and Options

The following table summarizes this behavior:

Return Code	-s Option Set	+s Option Set
0 (success)	execution continues	execution continues
1 (0 rows selected)	execution continues	execution continues
2 (warning)	execution continues	execution continues
4 (DB2 or SQL error)	execution stops	execution continues
8 (System error)	execution stops	execution stops

Statement Termination Character Option (-t):

The `-t` option tells the command line processor to use a semicolon (;) as the statement termination character, and disables the backslash (\) line continuation character.

Note: This option cannot be changed from within the interactive mode.

The default setting for this command option is OFF (+t or -t-).

To define a termination character, use `-td` followed by the chosen termination character. For example, `-tdx` sets `x` as the statement termination character.

The termination character cannot be used to concatenate multiple statements from the command line, since only the last non-blank character on each input line is checked for a termination symbol.

The statement termination character option does not affect any other command line processor option.

Verbose Output Option (-v):

The `-v` option causes the command line processor to echo (to standard output) the command text entered by the user prior to displaying the output, and any messages from that command. "ECHO" on page 160 is exempt from this option.

The default setting for this command option is OFF (+v or -v-).

The `-v` option has no effect if `+o` (or `-o-`) is specified.

The verbose output option does not affect any other command line processor option.

Show Warning Messages Option (-w):

The `-w` option tells the command line processor to show SQL statement warning messages.

The default setting for this command option is ON.

Save all Output to File Option (-z):

The `-zfilename` option causes all output generated by a command to be written to a specified file, and is useful for capturing a report that would otherwise scroll off the screen. It is similar to the `-r` option; in this case, however, messages, error codes, and other informational output are also

Command Line Processor Invocation and Options

written to the file. *Filename* is an absolute or relative file name which may include the directory path to the file. If the directory path is not specified, the current directory is used. New report entries are appended to the file.

The default setting for this command option is OFF (+z or -z-).

If the -a option is specified, SQLCA data is written to the file.

The -z option does not affect the -e option. If the -e option is specified, SQLCODE or SQLSTATE is written to standard output, not to a file.

If -z*filename* is set in **DB2OPTIONS**, the user can set the +z (or -z-) option from the command line to prevent output data for a particular command invocation from being written to the file.

The save all output to file option does not affect any other command line processor option.

Return Codes

When the command line processor finishes processing a command or an SQL statement, it returns an exit (or return) code. These codes are transparent to users executing CLP functions from the command line, but they can be retrieved when those functions are executed from a shell script.

For example, the following Bourne shell script executes the GET DATABASE MANAGER CONFIGURATION command, then inspects the CLP return code:

```
db2 get database manager configuration
if ["$?" = "0"]
then echo "OK!"
fi
```

The return code can be one of the following:

Code	Description
0	DB2 command or SQL statement executed successfully
1	SELECT or FETCH statement returned no rows
2	DB2 command or SQL statement warning
4	DB2 command or SQL statement error
8	Command line processor system error

The command line processor does not provide a return code while a user is executing statements from interactive mode, or while input is being read from a file (using the -f option).

A return code is available only after the user quits interactive mode, or when processing of an input file ends. In these cases, the return code is the logical OR of the distinct codes returned from the individual commands or statements executed to that point.

For example, if a user in interactive mode issues commands resulting in return codes of 0, 1, and 2, a return code of 3 will be returned after the user quits interactive mode. The individual codes 0, 1, and 2 are not returned. Return code 3 tells the user that

Using the Command Line Processor

during interactive mode processing, one or more commands returned a 1, and one or more commands returned a 2.

A return code of 4 results from a negative SQLCODE returned by a DB2 command or an SQL statement. A return code of 8 results only if the command line processor encounters a system error.

If commands are issued from an input file or in interactive mode, and the command line processor experiences a system error (return code 8), command execution is halted immediately. If one or more DB2 commands or SQL statements end in error (return code 4), command execution stops if the `-s` (Stop Execution on Command Error) option is set; otherwise, execution continues.

Using the Command Line Processor

The command line processor operates as follows:

- The CLP command (in either case) is typed at the command prompt.
- The command is sent to the command shell by pressing the ENTER key.
- Output is automatically directed to the standard output device.
- Piping and redirection are supported.
- The user is notified of successful and unsuccessful completion.
- Following execution of the command, control returns to the operating system command prompt, and the user may enter more commands.

Before accessing a database, the user must perform preliminary tasks, such as starting DB2 with “START DATABASE MANAGER” on page 390. The user must also connect to a database before it can be queried. Connect to a database by doing one of the following:

- Issue the SQL `CONNECT TO database` statement
- Establish an implicit connection to the default database defined by the environment variable **DB2DBDFT**.

For information about working with tables within a database, see the *SQL Reference*.

If a command exceeds the character limit allowed at the command prompt, a backslash (`\`) can be used as the line continuation character. When the command line processor encounters the line continuation character, it reads the next line and concatenates the characters contained on both lines. Alternatively, the `-t` option can be used to set a line termination character (see page 76). In this case, the line continuation character is invalid, and all statements and commands must end with the line termination character.

The command line processor recognizes a string called NULL as a null string. Fields that have been set previously to some value can later be set to null. For example,

```
db2 update database manager configuration using tm_database NULL
```

Using the Command Line Processor

sets the *tm_database* field to null. This operation is case sensitive. A lowercase null is not interpreted as a null string, but rather as a string containing the letters null.

Using the Command Line Processor in Command Files

CLP requests to the database manager can be imbedded in a shell script command file. The following example shows how to enter the CREATE TABLE statement in a shell script command file:

```
db2 "create table mytable (name VARCHAR(20), color CHAR(10))"
```

For more information about commands and command files, see the appropriate operating system manual.

Command Line Processor Design

The command line processor consists of two processes: the front-end process (the DB2 command), which acts as the user interface, and the back-end process (db2bp), which maintains a database connection.

Maintaining Database Connections

Each time that **db2** is invoked, a new front-end process is started. The back-end process is started by the first **db2** invocation, and can be explicitly terminated with "TERMINATE" on page 398. All front-end processes with the same parent are serviced by a single back-end process, and therefore share a single database connection.

For example, the following **db2** calls from the same operating system command prompt result in separate front-end processes sharing a single back-end process, which holds a database connection throughout:

```
db2 'connect to sample',  
db2 'select * from org',  
. foo (where foo is a shell script containing DB2 commands), and  
db2 -tf myfile.clp.
```

The following invocations from the same operating system prompt result in separate database connections because each has a distinct parent process, and therefore a distinct back-end process:

```
foo  
. foo &  
foo &  
sh foo
```

Communication between Front-end and Back-end Processes

The front-end process and back-end processes communicate through three message queues: a request queue, an input queue, and an output queue.

Using the Command Line Processor

Environment Variables

The following environment variables offer a means of configuring communication between the two processes:

Variable	Minimum	Maximum	Default
DB2BQTIME	1 second	5294967295	1 second
DB2BQTRY	0 tries	5294967295	60 tries
DB2RQTIME	1 second	5294967295	5 seconds
DB2IQTIME	1 second	5294967295	5 seconds

DB2BQTIME When the command line processor is invoked, the front-end process checks if the back-end process is already active. If it is active, the front-end process re-establishes a connection to it. If it is not active, the front-end process activates it. The front-end process then idles for the duration specified by the **DB2BQTIME** variable, and checks again. The front-end process continues to check for the number of times specified by the **DB2BQTRY** variable, after which, if the back-end process is still not active, it times out and returns an error message.

DB2BQTRY works in conjunction with the **DB2BQTIME** variable, and specifies the number of times the front-end process tries to determine whether the back-end process is active.

The values of **DB2BQTIME** and **DB2BQTRY** can be increased during peak periods to optimize query time.

DB2RQTIME Once the back-end process has been started, it waits on its request queue for a request from the front-end. It also waits on the request queue between requests initiated from the command prompt.

The **DB2RQTIME** variable specifies the length of time the back-end process waits for a request from the front-end process. At the end of this time, if no request is present on the request queue, the back-end process checks whether the parent of the front-end process still exists, and terminates itself if it does not exist. Otherwise, it continues to wait on the request queue.

DB2IQTIME When the back-end process receives a request from the front-end process, it sends an acknowledgement to the front-end process indicating that it is ready to receive input via the input queue. The back-end process then waits on its input queue. It also waits on the input queue while a batch file (specified with the `-f` option) is executing, and while the user is in interactive mode.

The **DB2IQTIME** variable specifies the length of time the back-end process waits on the input queue for the front-end process to pass the commands. After this time has elapsed, the back-end process checks whether the front-end process is active, and returns to wait on the request queue if the

Using the Command Line Processor

front-end process no longer exists. Otherwise, the back-end process continues to wait for input from the front-end process.

To view the values of these environment variables, use "LIST COMMAND OPTIONS" on page 238.

The back-end environment variables inherit the values set by the front-end process at the time the back-end process is initiated. However, if the front-end environment variables are changed, the back-end process will not inherit these changes. The back-end process must first be terminated, and then restarted (by issuing the **db2** command) to inherit the changed values.

An example of when the back-end process must be terminated is provided by the following scenario:

1. User A logs on, issues some CLP commands, and then logs off without issuing **TERMINATE**.
2. User B logs on using the same window.
3. When user B issues certain CLP commands, they fail with message DB21016 (system error).

The back-end process started by user A is still active when user B starts using the CLP, because the parent of user B's front-end process (the operating system window from which the commands are issued) is still active. The back-end process attempts to service the new commands issued by user B; however, user B's front-end process does not have enough authority to use the message queues of the back-end process, because it needs the authority of user A, who created that back-end process. A CLP session must end with a **TERMINATE** command before a user starts a new CLP session using the same operating system window. This creates a fresh back-end process for each new user, preventing authority problems, and setting the correct values of environment variables (such as **DB2INSTANCE**) in the new user's back-end process.

Usage Notes

Commands can be entered either in uppercase or in lowercase from the command prompt. However, parameters that are case sensitive to DB2 must be entered in the exact case desired. For example, the *comment-string* in the **WITH** clause of the **CHANGE DATABASE COMMENT** command is a case sensitive parameter.

Delimited identifiers are allowed in SQL statements. For more detailed information on the use of delimited identifiers in SQL statements, see the *SQL Reference*.

Special characters, or metacharacters (such as \$ & * () ; < > ? \ ' ") are allowed within CLP commands. If they are used outside the CLP interactive mode, or the CLP batch input mode, these characters are interpreted by the operating system shell. Quotation marks or an escape character are required if the shell is not to take any special action.

For example, when executed inside an AIX Korn shell environment,

```
db2 select * from org where division > 'Eastern'
```

Using the Command Line Processor

is interpreted as "select <the names of all files> from org where division". The result, an SQL syntax error, is redirected to the file Eastern. The following syntax produces the correct output:

```
db2 "select * from org where division > 'Eastern'"
```

Special characters vary from platform to platform. In the AIX Korn shell, the above example could be rewritten using an escape character (\), such as *, \>, or \'. In the OS/2 shell, * or \' results in a syntax error.

Most operating system environments allow input and output to be redirected. For example, if a connection to the SAMPLE database has been made, the following request queries the STAFF table, and sends the output to a file named staflist.txt in the mydata directory:

```
db2 "select * from staff" > mydata/staflist.txt
```

For environments such as Windows 3.1, where output redirection is not supported, CLP options can be used. For example, the request can be rewritten as

```
db2 -r mydata\staflist.txt "select * from staff"
```

```
db2 -z mydata\staflist.txt "select * from staff"
```

For more information on CLP options for Windows, see the *Installing and Using DB2 Clients for Windows* book.

The command line processor is not a programming language. For example, it does not support host variables, and the statement,

```
db2 connect to :HostVar in share mode
```

is syntactically incorrect, because :HostVar is not a valid database name.

The command line processor represents SQL NULL values as hyphens (-). If the column is numeric, the hyphen is placed at the right of the column. If the column is not numeric, the hyphen is at the left. For information about using the command line processor with DB2 Connect and host databases, see the *DB2 Connect User's Guide*.

Chapter 3. CLP Commands

This chapter describes the DB2 commands in alphabetical order. These commands are used to control the system interactively.

Note: Slashes (/) in directory paths are specific to UNIX based systems, and are equivalent to back slashes (\) in directory paths on OS/2 and Windows operating systems.

For information about how the command descriptions are organized, see “How the Command Descriptions are Organized” on page 1.

DB2 CLP Commands

The following table lists the CLP commands grouped by functional category:

<i>Table 4 (Page 1 of 4). DB2 CLP Commands</i>
CLP Session Control
“LIST COMMAND OPTIONS” on page 238
“UPDATE COMMAND OPTIONS” on page 409
“CHANGE ISOLATION LEVEL” on page 141
“SET RUNTIME DEGREE” on page 385
“TERMINATE” on page 398
“QUIT” on page 331
Database Manager Control
“START DATABASE MANAGER” on page 390
“STOP DATABASE MANAGER” on page 395
“GET DATABASE MANAGER CONFIGURATION” on page 188
“RESET DATABASE MANAGER CONFIGURATION” on page 356
“UPDATE DATABASE MANAGER CONFIGURATION” on page 413
“GET ADMIN CONFIGURATION” on page 171
“RESET ADMIN CONFIGURATION” on page 352
“UPDATE ADMIN CONFIGURATION” on page 405
Database Control
“RESTART DATABASE” on page 360
“CREATE DATABASE” on page 143
“DROP DATABASE” on page 157
“MIGRATE DATABASE” on page 303
“LIST INDOUBT TRANSACTIONS” on page 254
“ACTIVATE DATABASE” on page 87

DB2 CLP Commands

<i>Table 4 (Page 2 of 4). DB2 CLP Commands</i>
"DEACTIVATE DATABASE" on page 149
"LIST DRDA INDOUBT TRANSACTIONS" on page 249
Database Directory Management
"CATALOG DATABASE" on page 118
"UNCATALOG DATABASE" on page 399
"CATALOG DCS DATABASE" on page 121
"UNCATALOG DCS DATABASE" on page 401
"CHANGE DATABASE COMMENT" on page 139
"LIST DATABASE DIRECTORY" on page 240
"LIST DCS DIRECTORY" on page 247
"CATALOG GLOBAL DATABASE" on page 124
ODBC Management
"CATALOG ODBC DATA SOURCE" on page 135
"LIST ODBC DATA SOURCES" on page 264
"UNCATALOG ODBC DATA SOURCE" on page 404
"GET CLI CONFIGURATION" on page 176
"UPDATE CLI CONFIGURATION" on page 407
Client/Server Directory Management
"CATALOG LOCAL NODE" on page 129
"CATALOG NAMED PIPE NODE" on page 131
"CATALOG APPC NODE" on page 111
"CATALOG APPCLU NODE" on page 114
"CATALOG APPN NODE" on page 116
"CATALOG IPX/SPX NODE" on page 126
"CATALOG NETBIOS NODE" on page 133
"CATALOG TCP/IP NODE" on page 136
"UNCATALOG NODE" on page 402
"LIST NODE DIRECTORY" on page 258
Network Support
"REGISTER" on page 340
"DEREGISTER" on page 151
Database Configuration
"GET DATABASE CONFIGURATION" on page 179
"RESET DATABASE CONFIGURATION" on page 354
"UPDATE DATABASE CONFIGURATION" on page 411
Recovery

DB2 CLP Commands

<i>Table 4 (Page 3 of 4). DB2 CLP Commands</i>
"BACKUP DATABASE" on page 93
"RECONCILE" on page 335
"RESTORE DATABASE" on page 362
"ROLLFORWARD DATABASE" on page 370
"LIST HISTORY" on page 251
"PRUNE HISTORY" on page 325
"UPDATE RECOVERY HISTORY FILE" on page 417
Operational Utilities
"FORCE APPLICATION" on page 169
"LIST PACKAGES/TABLES" on page 266
"REORGCHK" on page 345
"REORGANIZE TABLE" on page 342
"RUNSTATS" on page 378
Database Monitoring
"GET MONITOR SWITCHES" on page 201
"UPDATE MONITOR SWITCHES" on page 415
"GET DATABASE MANAGER MONITOR SWITCHES" on page 198
"GET SNAPSHOT" on page 203
"RESET MONITOR" on page 358
"LIST ACTIVE DATABASES" on page 235
"LIST APPLICATIONS" on page 236
"LIST DCS APPLICATIONS" on page 244
Data Utilities
"EXPORT" on page 161
"IMPORT" on page 219
"LOAD" on page 276
"LOAD QUERY" on page 301
Application Preparation
"PRECOMPILE PROGRAM" on page 305
"BIND" on page 98
"REBIND" on page 332
Remote Server Utilities
"ATTACH" on page 91
"DETACH" on page 156
Table Space Management
"LIST TABLESPACE CONTAINERS" on page 269

DB2 CLP Commands

<i>Table 4 (Page 4 of 4). DB2 CLP Commands</i>
"SET TABLESPACE CONTAINERS" on page 387
"LIST TABLESPACES" on page 271
"QUIESCE TABLESPACES FOR TABLE" on page 328
Node Management
"ADD NODE" on page 89
"DROP NODE VERIFY" on page 159
"LIST NODES" on page 263
Nodegroup Management
"LIST NODEGROUPS" on page 261
"REDISTRIBUTE NODEGROUP" on page 337
Additional Commands
"DESCRIBE" on page 152
"ECHO" on page 160
"GET AUTHORIZATIONS" on page 174
"GET CONNECTION STATE" on page 178
"GET INSTANCE" on page 200
"HELP" on page 217
"INVOKE STORED PROCEDURE" on page 233
"QUERY CLIENT" on page 326
"SET CLIENT" on page 382
"INITIALIZE TAPE" on page 232
"REWIND TAPE" on page 369
"SET TAPE POSITION" on page 389

ACTIVATE DATABASE

Activates the specified database and starts up all necessary database services, so that the database is available for connection and use by any application.

Scope

This command activates the specified database on all nodes within the system. If one or more of these nodes encounters an error during activation of the database, a warning is returned. The database remains activated on all nodes on which the command has succeeded.

Authorization

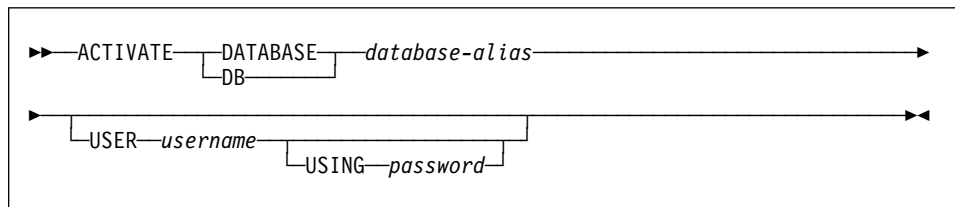
One of the following:

sysadm
sysctrl
sysmaint

Required Connection

None

Command Syntax



Command Parameters

database-alias

Specifies the alias of the database to be started.

USER *username*

Specifies the user starting the database.

USING *password*

Specifies the password for the user name.

Usage Notes

If a database has not been started, and a CONNECT TO (or an implicit connect) is issued in an application, the application must wait while the database manager starts the required database, before it can do any work with that database. However, once the database is started, other applications can simply connect and use it without spending time on its start up.

ACTIVATE DATABASE

Database administrators can use `ACTIVATE DATABASE` to start up selected databases. This eliminates any application time spent on database initialization.

Databases initialized by `ACTIVATE DATABASE` can be shut down by “`DEACTIVATE DATABASE`” on page 149, or by “`STOP DATABASE MANAGER`” on page 395.

If a database was started by a `CONNECT TO` (or an implicit connect) and subsequently an `ACTIVATE DATABASE` is issued for that same database, then `DEACTIVATE DATABASE` must be used to shut down that database. If `ACTIVATE DATABASE` was not used to start the database, the database will shut down when the last application disconnects.

`ACTIVATE DATABASE` behaves in a similar manner to a `CONNECT TO` (or an implicit connect) when working with a database requiring a restart (for example, database in an inconsistent state). The database will be restarted before it can be initialized by `ACTIVATE DATABASE`. Restart will only be performed if the database is configured to have `AUTORESTART ON`.

Note: The application issuing the `ACTIVATE DATABASE` command cannot have an active database connection to any database.

See Also

“`DEACTIVATE DATABASE`” on page 149

“`STOP DATABASE MANAGER`” on page 395.

ADD NODE

Adds a new node to the parallel database system. This command creates database partitions for all databases currently defined in the MPP server on the new node. The user can specify the source node for any temporary table spaces to be created with the databases, or specify that no temporary table spaces are to be created. The command must be issued from the node that is being added, and can only be issued on an MPP server.

Scope

This command only affects the node on which it is executed.

Authorization

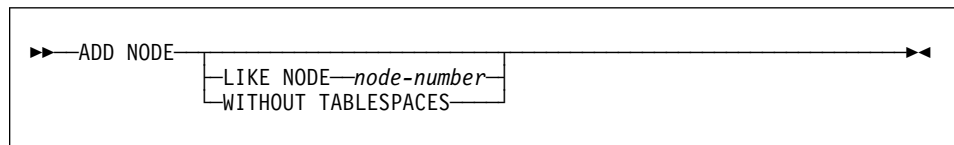
One of the following:

sysadm
sysctrl

Required Connection

None

Command Syntax



Command Parameters

LIKE NODE *node-number*

Specifies that the containers for the temporary table spaces will be the same as the containers on the specified *node-number* for each database in the instance. The node specified must be a node that is already in the `db2nodes.cfg` file.

WITHOUT TABLESPACES

Specifies that containers for the temporary table spaces are not created for any of the databases. The `ALTER TABLESPACE` statement must be used to add temporary table space containers to each database before the database can be used.

Note: If no option is specified, containers for the temporary table spaces will be the same as the containers on the catalog node for each database. The catalog node may be a different node for each database in the MPP system.

ADD NODE

Usage Notes

Before adding a new node, ensure that there is sufficient storage for the containers that must be created for all existing databases on the system.

The add node operation creates an empty database partition on the new node for every database that exists in the instance. The configuration parameters for the new database partitions are set to the default value.

If an add node operation fails while creating a database partition locally, it enters a clean-up phase, in which it locally drops all databases that have been created. This means that the database partitions are removed only from the node being added (that is, the local node). Existing database partitions remain unaffected on all other nodes. If this fails, no further clean up is done, and an error is returned.

The database partitions on the new node cannot be used to contain user data until after the ALTER NODEGROUP statement has been used to add the node to a nodegroup. For details, see the *SQL Reference*.

This command will fail if a create database or a drop database operation is in progress. The command can be reissued once the operation has completed.

If temporary table spaces are to be created with the database partitions, ADD NODE may have to communicate with another node in the MPP system in order to retrieve the table space definitions. The *start_stop_time* database manager configuration parameter is used to specify the time, in minutes, by which the other node must respond with the table space definitions. If this time is exceeded, the command fails. Increase the value of *start_stop_time*, and reissue the command.

See Also

“START DATABASE MANAGER” on page 390.

ATTACH

Enables an application to specify the instance at which instance-level commands (CREATE DATABASE and FORCE APPLICATION, for example) are to be executed. This instance may be the current instance, another instance on the same workstation, or an instance on a remote workstation.

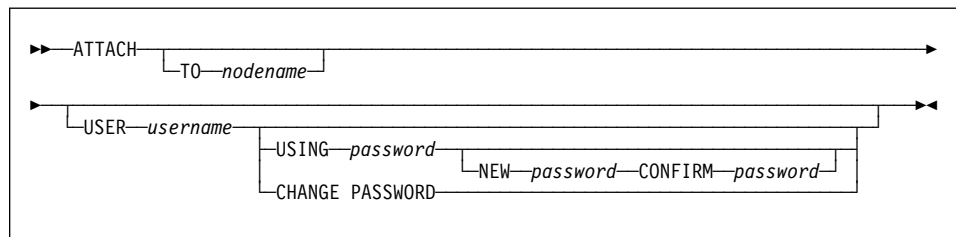
Authorization

None

Required Connection

None. This command establishes an instance attachment.

Command Syntax



Command Parameters

TO *nodename*

Alias of the instance to which the user wants to attach. This instance must have a matching entry in the local node directory. The only exception to this is the local instance (as specified by the **DB2INSTANCE** environment variable) which may be specified as the object of an attach, but which cannot be used as a node name in the node directory.

USER *username*

Specifies the authentication identifier.

USING *password*

Specifies the password for the user name. If a user name is specified, but a password is *not* specified, the user is prompted for the current password. The password is not displayed at entry.

NEW *password*

Specifies the new password that is to be assigned to the user name. Passwords can be up to 18 characters in length. The system on which the password will be changed depends on how user authentication has been set up.

CONFIRM *password*

A string that must be identical to the new password. This parameter is used to catch entry errors.

ATTACH

CHANGE PASSWORD

If this option is specified, the user is prompted for the current password, a new password, and for confirmation of the new password. Passwords are not displayed at entry.

Example

Catalog two remote nodes:

```
db2 catalog tcpip node node1 remote freedom server server1
db2 catalog tcpip node node2 remote flash server server1
```

Attach to the first node, force all users, and then detach:

```
db2 attach to node1
db2 force application all
db2 detach
```

Attach to the second node, and see who is on:

```
db2 attach to node2
db2 list applications
```

After the command returns agent IDs 1, 2 and 3, force 1 and 3, and then detach:

```
db2 force application (1, 3)
db2 detach
```

Attach to the current instance (not necessary, will be implicit), force all users, then detach (AIX only):

```
db2 attach to $DB2INSTANCE
db2 force application all
db2 detach
```

Usage Notes

If *nodename* is omitted from the command, information about the current state of attachment is returned.

If ATTACH has not been executed, instance-level commands are executed against the current instance, specified by the **DB2INSTANCE** environment variable.

See Also

“DETACH” on page 156.

BACKUP DATABASE

Creates a backup copy of a database or a table space.

Scope

This command only affects the node on which it is executed.

Authorization

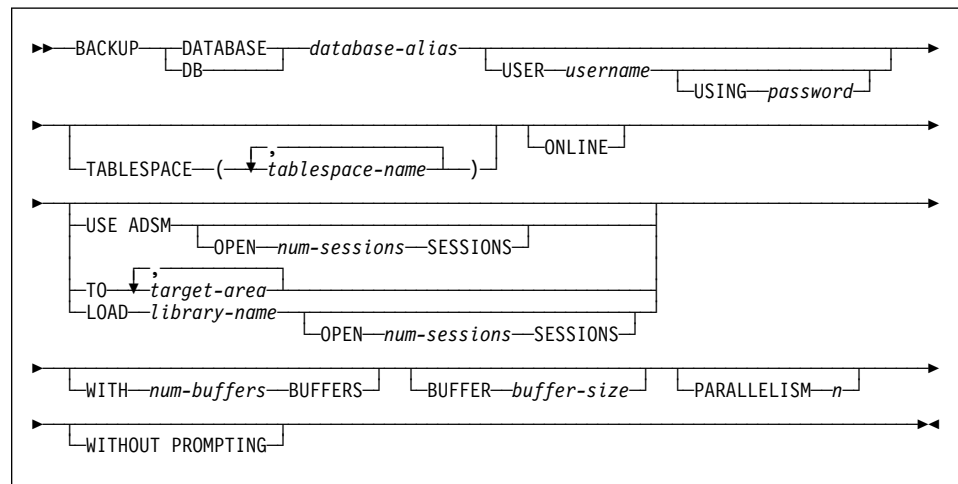
One of the following:

sysadm
sysctrl
sysmaint

Required Connection

Database. This command automatically establishes a connection to the specified database.

Command Syntax



Command Parameters

DATABASE *database-alias*

Specifies the alias of the database to back up.

USER *username*

Identifies the user name under which to back up the database.

USING *password*

The password used to authenticate the user name. If the password is omitted, the user is prompted to enter it.

BACKUP DATABASE

TABLESPACE *tablespace-name*

A list of names used to specify the table spaces to be backed up.

ONLINE

Specifies online backup. The default is offline backup.

USE ADSM

Specifies that the backup is to use ADSM managed output.

OPEN *num-sessions* **SESSIONS**

The number of I/O sessions to be used with ADSM or another product.

TO *target-area*

A list of directory or tape device names. The full path on which the directory resides must be specified. The target must reside on the database server. This parameter may be repeated to specify the target directories and devices that the backup image will span. If more than one target is specified (target1, target2, and target3, for example), target1 will be opened first. The media header and special files (including the configuration file, table space table, and history file) are placed in target1. All remaining targets are opened, and are then used in parallel during the backup.

Use of tape devices or floppy disks may generate messages and prompts for user input. Valid response options are:

- c** Continue. Continue using the device that generated the warning message (for example, when a new tape has been mounted)
- d** Device terminate. Stop using *only* the device that generated the warning message (for example, when there are no more tapes)
- t** Terminate. Abort the backup or restore utility.

Tape is not supported on OS/2. On OS/2, 0 or 0: can be specified to cause the backup operation to call the user exit program (see the *Administration Guide*). This option is not valid on any other platform.

LOAD *library-name*

The name of the shared library (DLL on OS/2 or the Windows operating system) containing the vendor backup and restore I/O functions to be used. It can contain the full path. If the full path is not given, it will default to the path on which the user exit program resides.

WITH *num-buffers* **BUFFERS**

The number of buffers to be used.

BUFFER *buffer-size*

The size, in pages, of the buffer used when building the backup image. The minimum value for this parameter is 16 pages; the default value is 1024 pages. If a buffer-size of 0 is specified, the value of the database manager configuration parameter *backbufsz* is used.

PARALLELISM *n*

Specifies the number of buffer manipulators to be spawned during the restore process. The default value is 1.

BACKUP DATABASE

WITHOUT PROMPTING

Specifies that the backup will run unattended, and that any actions which normally require user intervention will return an error message.

Examples

```
db2 backup database sample use adsm open 2 sessions with 4 buffers
```

```
db2 backup database payroll tablespace syscatspace, userspace1 to  
/dev/rmt0, /dev/rmt1 with 8 buffers without prompting
```

Usage Notes

Database Level Backup

Note: Backup each database on a regular basis. If a database becomes damaged or corrupted, it can be returned to the state of the backed-up copy (see “RESTORE DATABASE” on page 362).

If a successfully restored database was enabled for roll-forward recovery at the time of the backup operation, it can be returned to the state it was in prior to the occurrence of damage (see “ROLLFORWARD DATABASE” on page 370).

The backup can be directed to fixed disk, diskette, tape, ADSTM utility, or to other vendor products enabled for DB2.

On UNIX based systems, a backup file name consists of the concatenation of several units of information separated by periods:

dbname.type.instance.nodexxxx.catnxxxx.yyyymmddhhmmss.seq

dbname	1 to 8 character database alias.
type	Type of backup taken (0 for full database, or 3 for table space level backup).
instance	1 to 8 character database instance name.
nodexxxx	The number of the node. In non-partitioned database systems, this is always zero (NODE0000). In a partitioned database system, it is NODExxxx, where xxxx is the number assigned to the node in the db2nodes.cfg file.
catnxxxx	The number of the catalog node for the database. In non-partitioned database systems, this is always zero (CATN0000). In a partitioned database system, it is CATNxxxx, where xxxx is the number assigned to the node in the db2nodes.cfg file.
yyymmdd	Date (year month day).
hhmmss	Time (hour minute second).
seq	A file extension consisting of a 3-digit sequence number.

BACKUP DATABASE

In addition to fixed disk, tape, ADSM and other vendors, the backup may be directed to diskettes. Since there is no general tape support on OS/2 or the Windows operating system, each type of tape device requires a unique device driver.

To back up to the FAT file system on OS/2 or the Windows operating system, users must conform to the 8.3 naming restriction. The file is placed in a 5-level subdirectory tree as follows:

dbname.type\db2instance.nodexxx\catnxxx\yyyymmdd\hhmmss.seq

dbname	1 to 8 character database alias.
type	Type of backup taken. 0 for full database, 3 for table space level backup, 4 for copy from a table load.
db2instance	1 to 8 character database instance name.
nodexxx	The number of the node. In non-partitioned database systems, this is always zero (NODE000). In a partitioned database system, it is NODExxx, where xxx is the number assigned to the node in the db2nodes.cfg file.
catnxxx	The number of the catalog node for the database. In non-partitioned database systems, this is always zero (CATN000). In a partitioned database system, it is CATNxxx, where xxx is the number assigned to the node in the db2nodes.cfg file.
yyyymmdd	Date (year month day).
hhmmss	Time (hour minute second).
seq	A file extension consisting of a 3-digit sequence number.

Online backups are permitted only if roll-forward recovery is enabled. The utility can perform an online backup while it is being accessed and modified by other applications.

To perform an offline backup, the utility must be able to connect to the database in exclusive mode. BACKUP fails if any application, including the calling application, is already connected to the database. If the connection is successful, BACKUP locks out other applications until the backup is completed.

Execute BACKUP DATABASE offline when the database is not currently needed.

An offline backup operation will fail if the database is not in a consistent state. If the database is inconsistent, it must be restarted to be brought back to a consistent state through crash recovery before BACKUP is executed (see "RESTART DATABASE" on page 360, and a description of the *autorestart* configuration parameter in the *Administration Guide*). If the database is in a partially restored state after a system failure during restoration, the restore operation must be successfully rerun before a backup can be executed. If the database has been restored and a roll-forward operation is needed, the database must be rolled forward to a consistent state before it can be backed up.

BACKUP DATABASE

If a database is changed from roll-forward disabled to roll-forward enabled state, either the *logretain* or the *userexit* database configuration options must be enabled before a backup of the database can be made (see “GET DATABASE CONFIGURATION” on page 179).

Table Space Level Backup

A table space level backup contains one or more table spaces for a database, specified when BACKUP is executed. A table space level backup can be taken online or offline.

Table space level backup can be used to recover from problems that only affect specific table spaces. While this recovery is taking place, all other table spaces are available for processing.

To ensure that restored table spaces are synchronized with the rest of the database, the table spaces must be rolled forward to the end of the log (or to the point where the table spaces were last used). For this reason, table space level backup and restore can only be performed if roll-forward recovery is enabled. If roll-forward recovery is disabled at any time after a table space level backup is executed, it will not be possible to restore from the backup, and then to roll the table space forward to the current point in time. In this case, all table space level backups taken prior to that time are no longer restorable. The restore operation will fail if the user tries to restore from such a backup. In cases where it cannot be determined that the backup is not valid (if, for instance, the database has been restored and rolled forward, thus creating a new log sequence), the restore may be successful, and the broken restore set will be detected during roll-forward recovery.

The user may choose to separate data, index, long field (LONG), and large objects (LOB) into different table spaces. Long field and LOB data for the same table must reside in the same table space.

Each component of a table may be backed up and restored with the table space in which it resides, independently of the other components of the table.

It is not necessary to back up table spaces for temporary tables. If a list of table spaces to be backed up contains such a table space, BACKUP fails.

Table space level backup and restore cannot be run concurrently.

See Also

“MIGRATE DATABASE” on page 303
“RESTORE DATABASE” on page 362
“ROLLFORWARD DATABASE” on page 370.

BIND

BIND

Invokes the bind utility, which prepares SQL statements stored in the bind file generated by the precompiler, and creates a package that is stored in the database.

Scope

This command can be issued from any node in `db2nodes.cfg`. It updates the database catalogs on the catalog node. Its effects are visible to all nodes.

Authorization

One of the following:

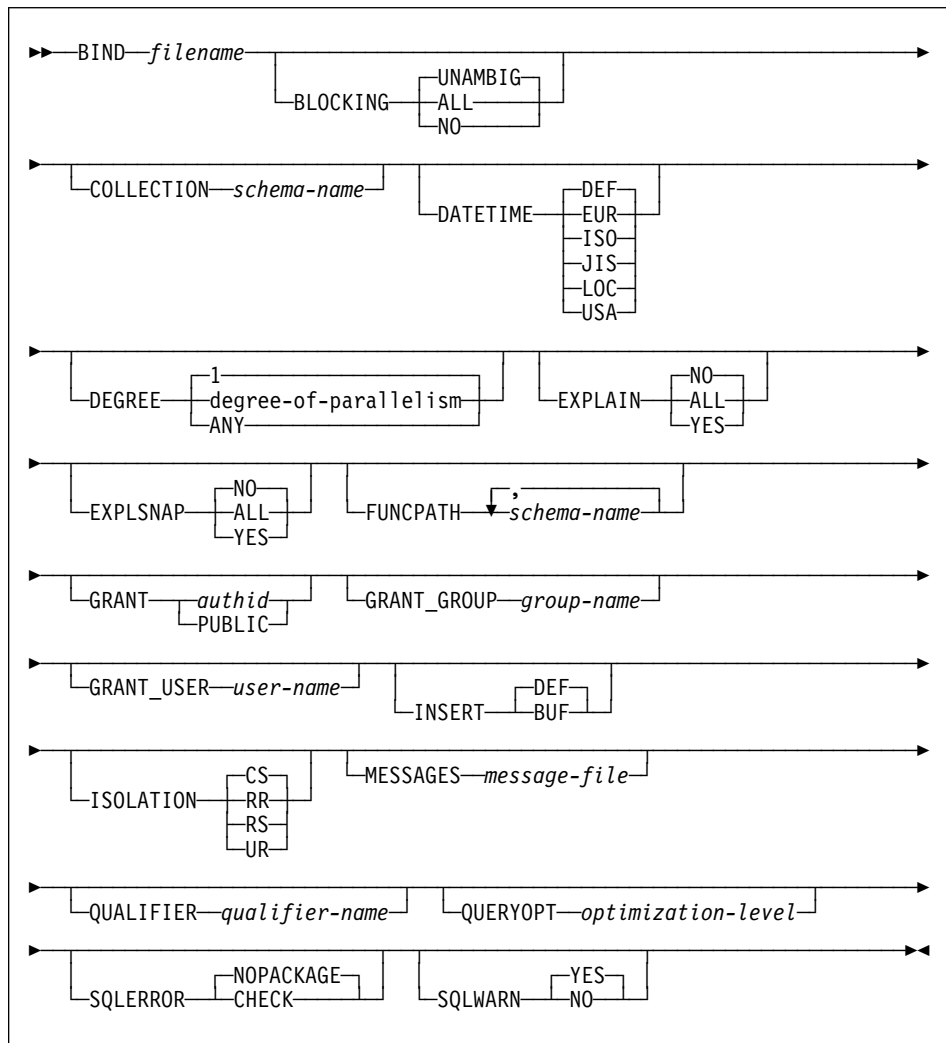
- *sysadm* or *dbadm* authority
- BINDADD privilege if a package does not exist and one of:
 - IMPLICIT_SCHEMA authority on the database if the schema name of the package does not exist
 - CREATEIN privilege on the schema if the schema name of the package exists
- ALTERIN privilege on the schema if the package exists
- BIND privilege on the package if it exists.

The user also needs all privileges required to compile any static SQL statements in the application. Privileges granted to groups are not used for authorization checking of static statements. If the user has *sysadm* authority, but not explicit privileges to complete the bind, the database manager grants explicit *dbadm* authority automatically.

Required Connection

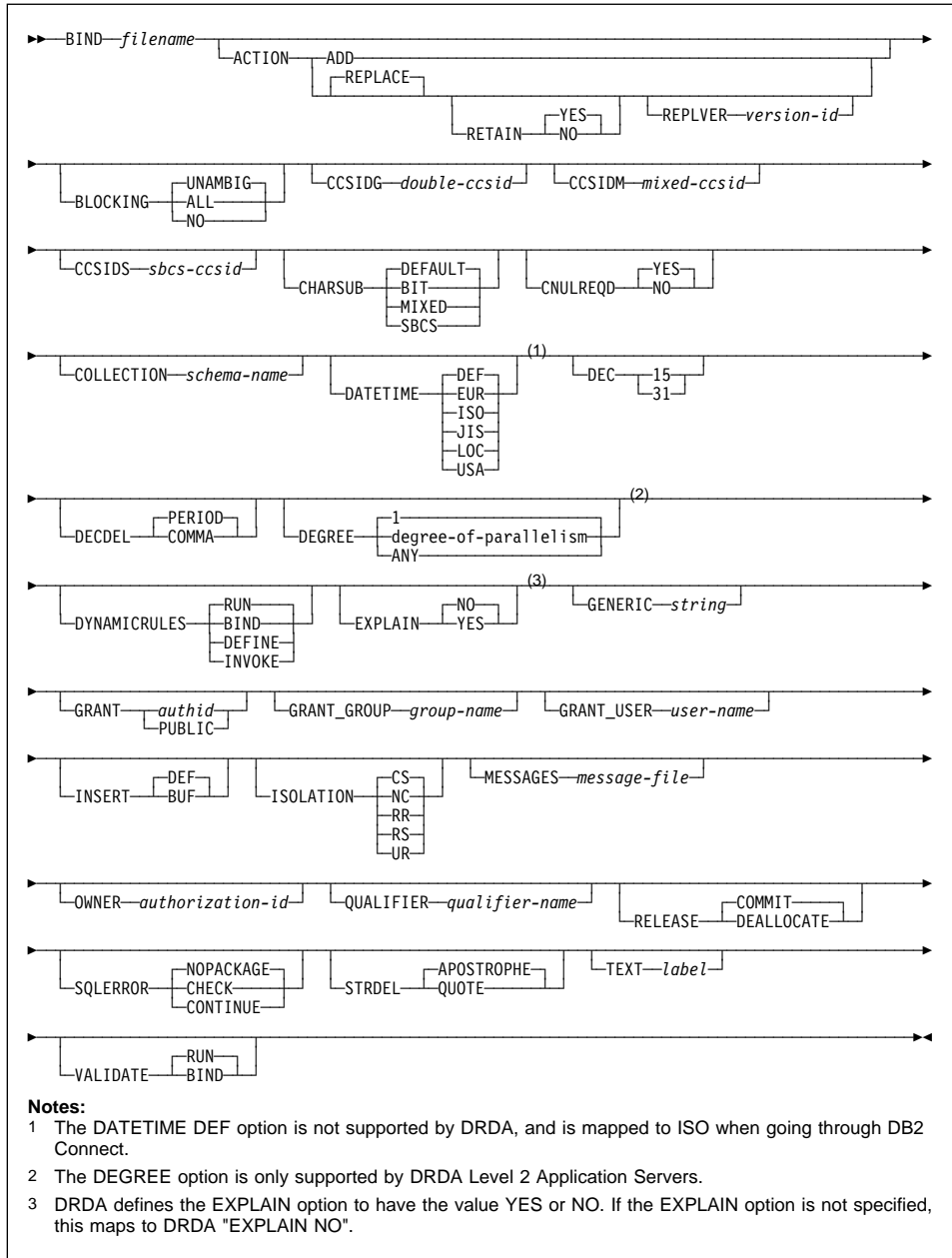
Database. If implicit connect is enabled, a connection to the default database is established.

Command Syntax
For DB2



BIND

For DRDA



Command Parameters

filename

Specifies the name of the bind file that was generated when the application program was precompiled, or a list file containing the names of several bind files. Bind files have the extension `.bnd`. The full path name can be specified.

If a list file is specified, the `@` character must be the first character of the list file name. The list file can contain several lines of bind file names. Bind files listed on the same line must be separated by plus (+) characters, but a + cannot appear in front of the first file listed on each line, or after the last bind file listed. For example,

```
/u/smith/sql1lib/bnd/@a11.lst
```

is a list file that contains the following bind files:

```
mybind1.bnd+mybind.bnd2+mybind3.bnd+
mybind4.bnd+mybind5.bnd+
mybind6.bnd+
mybind7.bnd
```

ACTION

Indicates whether the package can be added or replaced. This DRDA precompile/bind option is not supported by DB2.

ADD

Indicates that the named package does not exist, and that a new package is to be created. If the package already exists, execution stops, and a diagnostic error message is returned.

REPLACE

Indicates that the old package is to be replaced by a new one with the same location, collection, and package name.

RETAIN

Indicates whether EXECUTE authorities are to be preserved when a package is replaced. If ownership of the package changes, the new owner grants the BIND and EXECUTE authority to the previous package owner.

NO

Does not preserve EXECUTE authorities when a package is replaced.

YES

Preserves EXECUTE authorities when a package is replaced.

REPLVER version-id

Replaces a specific version of a package. The version identifier specifies which version of the

BIND

package is to be replaced. Maximum length is 254 characters.

BLOCKING

For information about row blocking, see the *Administration Guide* or the *Embedded SQL Programming Guide*.

ALL

Specifies to block for:

- Read-only cursors
- Cursors not specified as FOR UPDATE OF

Ambiguous cursors are treated as read-only.

NO

Specifies not to block any cursors. Ambiguous cursors are treated as updateable.

UNAMBIG

Specifies to block for:

- Read-only cursors
- Cursors not specified as FOR UPDATE OF

Ambiguous cursors are treated as updateable.

CCSIDG *double-ccsid*

An integer specifying the coded character set identifier (CCSID) to be used for double byte characters in character column definitions (without a specific CCSID clause) in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

CCSIDM *mixed-ccsid*

An integer specifying the coded character set identifier (CCSID) to be used for mixed byte characters in character column definitions (without a specific CCSID clause) in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

CCSIDS *sbcsc-ccsid*

An integer specifying the coded character set identifier (CCSID) to be used for single byte characters in character column definitions (without a specific CCSID clause) in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

CHARSUB

Designates the default character sub-type that is to be used for column definitions in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2.

BIT

Use the FOR BIT DATA SQL character sub-type in all new character columns for which an explicit sub-type is not specified.

DEFAULT

Use the target system defined default in all new character columns for which an explicit sub-type is not specified.

MIXED

Use the FOR MIXED DATA SQL character sub-type in all new character columns for which an explicit sub-type is not specified.

SBCS

Use the FOR SBCS DATA SQL character sub-type in all new character columns for which an explicit sub-type is not specified.

CNULREQD

This option is related to the **langlevel** precompile option, which is not supported by DRDA. It is valid only if the bind file is created from a C or a C++ application. This DRDA bind option is not supported by DB2.

NO

The application was coded on the basis of the **langlevel** SAA1 precompile option with respect to the null terminator in C string host variables.

YES

The application was coded on the basis of the **langlevel** MIA precompile option with respect to the null terminator in C string host variables.

COLLECTION *schema-name*

Specifies an 8-character collection identifier for the package. If not specified, the authorization identifier for the user processing the package is used.

DATETIME

Specifies the date and time format to be used. For more information about date and time formats, see the *SQL Reference*.

DEF

Use a date and time format associated with the country code of the database.

EUR

Use the IBM standard for Europe date and time format.

ISO

Use the date and time format of the International Standards Organization.

JIS

Use the date and time format of the Japanese Industrial Standard.

LOC

Use the date and time format in local form associated with the country code of the database.

USA

Use the IBM standard for U.S. date and time format.

BIND

DEC

Specifies the maximum precision to be used in decimal arithmetic operations. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

15

15-digit precision is used in decimal arithmetic operations.

31

31-digit precision is used in decimal arithmetic operations.

DECDEL

Designates whether a period (.) or a comma (,) will be used as the decimal point indicator in decimal and floating point literals. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

COMMA

Use a comma (,) as the decimal point indicator.

PERIOD

Use a period (.) as the decimal point indicator.

DEGREE

Specifies the degree of parallelism for the execution of static SQL statements in an SMP system. This option does not affect CREATE INDEX parallelism.

1

The execution of the statement will not use parallelism.

degree-of-parallelism

Specifies the degree of parallelism with which the statement can be executed, a value between 2 and 32 767 (inclusive).

ANY

Specifies that the execution of the statement can involve parallelism using a degree determined by the database manager.

DYNAMICRULES

Specifies which authorization identifier to use when dynamic SQL in a package is executed. This DRDA precompile/bind option is not supported by DB2.

BIND

Indicates that the authorization identifier used for the execution of dynamic SQL is the package owner.

RUN

Indicates that the authorization identifier used for the execution of dynamic SQL is the *authid* of the person executing the package.

DEFINE

Indicates that the authorization identifier used for the execution of dynamic SQL statements in a UDF or stored procedure is the definer of the UDF or stored procedure.

INVOKE

Indicates that the authorization identifier used for the execution of dynamic SQL statements in a UDF or stored procedure is the invoker of the UDF or stored procedure.

EXPLAIN

Stores information in the Explain tables about the access plans chosen for each SQL statement in the package. DRDA does not support the ALL value for this option.

NO

Explain information will not be captured.

YES

Explain tables will be populated with information about the chosen access plan.

ALL

Explain information for each eligible static SQL statement will be placed in the Explain tables. In addition, Explain information will be gathered for eligible dynamic SQL statements at run time, even if the CURRENT EXPLAIN SNAPSHOT register is set to NO. For more information about special registers, see the *SQL Reference*.

Note: This value for EXPLAIN is not supported by DRDA.

EXPLSNAP

Stores Explain Snapshot information in the Explain tables. This DB2 precompile/bind option is not supported by DRDA.

NO

An Explain Snapshot will not be captured.

YES

An Explain Snapshot for each eligible static SQL statement will be placed in the Explain tables.

ALL

An Explain Snapshot for each eligible static SQL statement will be placed in the Explain tables. In addition, Explain Snapshot information will be gathered for eligible dynamic SQL statements at run time, even if the CURRENT EXPLAIN SNAPSHOT register is set to NO. For more information about special registers, see the *SQL Reference*.

FUNCPATH

Specifies the function path to be used in resolving user-defined distinct types and functions in static SQL. If this option is not specified, the default function path is "SYSIBM","SYSFUN",USER where USER is the value of

BIND

the USER special register. This DB2 precompile/bind option is not supported by DRDA.

schema-name

A short SQL identifier, either ordinary or delimited, which identifies a schema that exists at the application server. No validation that the schema exists is made at precompile or at bind time. The same schema cannot appear more than once in the function path. The number of schemas that can be specified is limited by the length of the resulting function path, which cannot exceed 254 bytes. The schema SYSIBM does not need to be explicitly specified; it is implicitly assumed to be the first schema if it is not included in the function path. For more information, see the *SQL Reference*.

GENERIC *string*

Provides a means of passing new bind options to a target DRDA database. Supports the binding of new options that are defined in the target database, but that are not known to the local command. Do not use this option to pass bind options that *are* defined in "BIND" on page 98 or "PRECOMPILE PROGRAM" on page 305. This option can substantially improve dynamic SQL performance. The syntax is as follows:

```
generic "option1 value1 option2 value2 ..."
```

Each option and value must be separated by one or more blank spaces. For example, if the target DRDA database is DB2 MVS Version 5, one could use:

```
generic "keepdynamic yes"
```

to bind the new **keepdynamic** YES option, which is not defined locally on the PRECOMPILE PROGRAM or the BIND command.

The maximum length of the string is 1023 bytes. This DRDA bind option is currently only supported by DB2 MVS Version 5; it is not supported by DB2.

GRANT

authid

Grants EXECUTE and BIND privileges to a specified user name or group ID.

PUBLIC

Grants EXECUTE and BIND privileges to PUBLIC.

GRANT_GROUP *group-name*

Grants EXECUTE and BIND privileges to a specified group ID.

GRANT_USER *user-name*

Grants EXECUTE and BIND privileges to a specified user name.

INSERT

Allows a program being precompiled or bound against a DB2 Universal Database Extended Enterprise Edition server to request that data inserts be buffered to increase performance.

BUF

Specifies that inserts from an application should be buffered.

DEF

Specifies that inserts from an application should not be buffered.

ISOLATION

Determines how far a program bound to this package can be isolated from the effect of other executing programs. For more information about isolation levels, see the *SQL Reference*.

CS

Specifies Cursor Stability as the isolation level.

NC

No Commit. Specifies that commitment control is not to be used. This isolation level is not supported by DB2.

RR

Specifies Repeatable Read as the isolation level.

RS

Specifies Read Stability as the isolation level. Read Stability ensures that the execution of SQL statements in the package is isolated from other application processes for rows read and changed by the application.

UR

Specifies Uncommitted Read as the isolation level.

MESSAGES *message-file*

Specifies the destination for warning, error, and completion status messages. A message file is created whether the bind is successful or not. If a message file name is not specified, the messages are written to standard output. If the complete path to the file is not specified, the current directory is used. If the name of an existing file is specified, the contents of the file are overwritten.

OWNER *authorization-id*

Designates an 8-character authorization identifier for the package owner. The owner must have the privileges required to execute the SQL statements in the package. The default is the primary authorization ID of the precompile/bind process if this option has not been explicitly specified. This DRDA precompile/bind option is not supported by DB2.

QUALIFIER *qualifier-name*

Provides an 8-character implicit qualifier for unqualified objects contained in the package. The default is the owner's authorization ID, whether or not **owner** is explicitly specified.

BIND

QUERYOPT *optimization-level*

Indicates the desired level of optimization for all static SQL statements contained in the package. The default value is 5. For the complete range of optimization levels available, see the SET CURRENT QUERY OPTIMIZATION statement in the *SQL Reference*. This DB2 precompile/bind option is not supported by DRDA.

RELEASE

Indicates whether resources are released at each COMMIT point, or when the application terminates. This DRDA precompile/bind option is not supported by DB2.

COMMIT

Release resources at each COMMIT point. Used for dynamic SQL statements.

DEALLOCATE

Release resources only when the application terminates.

SQLERROR

Indicates whether to create a package or a bind file if an error is encountered.

CHECK

Specifies that the target system performs all syntax and semantic checks on the SQL statements being bound. A package will not be created as part of this process. If, while binding, an existing package with the same name and version is encountered, the existing package is neither dropped nor replaced even if **action replace** was specified.

CONTINUE

A package or a bind file is created even when SQL errors are encountered. This option is not supported by DB2.

NOPACKAGE

A package or a bind file is not created if an error is encountered.

SQLWARN

Indicates whether warnings will be returned from the compilation of dynamic SQL statements (via PREPARE or EXECUTE IMMEDIATE), or from describe processing (via PREPARE...INTO or DESCRIBE). This DB2 precompile/bind option is not supported by DRDA.

NO

Warnings will not be returned from the SQL compiler.

YES

Warnings will be returned from the SQL compiler.

Note: SQLCODE +238 is an exception. It is returned regardless of the **sqlwarn** option value.

STRDEL

Designates whether an apostrophe (') or double quotation marks (") will be used as the string delimiter within SQL statements. This DRDA

precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

APOSTROPHE

Use an apostrophe (') as the string delimiter.

QUOTE

Use double quotation marks (") as the string delimiter.

TEXT *label*

The description of a package. Maximum length is 255 characters. The default value is blanks. This DRDA precompile/bind option is not supported by DB2.

VALIDATE

Determines when the database manager checks for authorization errors and object not found errors. The package owner authorization ID is used for validity checking. This DRDA precompile/bind option is not supported by DB2.

BIND

Validation is performed at precompile/bind time. If all objects do not exist, or all authority is not held, error messages are produced. If **sqlerror continue** is specified, a package/bind file is produced despite the error message, but the statements in error are not executable.

RUN

Validation is attempted at bind time. If all objects exist, and all authority is held, no further checking is performed at execution time.

If all objects do not exist, or all authority is not held at precompile/bind time, warning messages are produced, and the package is successfully bound, regardless of the **sqlerror continue** option setting. However, authority checking and existence checking for SQL statements that failed these checks during the precompile/bind process may be redone at execution time.

Example

The following example binds `myapp.bnd` (the bind file generated when the `myapp.sqc` program was precompiled) to the database to which a connection has been established:

```
db2 bind myapp.bnd
```

Any messages resulting from the bind process are sent to standard output.

Usage Notes

Binding can be done as part of the precompile process for an application program source file, or as a separate step at a later time. Use **BIND** when binding is performed as a separate process.

BIND

The name used to create the package is stored in the bind file, and is based on the source file name from which it was generated (existing paths or extensions are discarded). For example, a precompiled source file called `myapp.sql` generates a default bind file called `myapp.bnd` and a default package name of `MYAPP`. However, the bind file name and the package name can be overridden at precompile time by using the **bindfile** and the **package** options.

Binding a package with a schema name that does not already exist will result in the implicit creation of that schema. The schema owner is `SYSIBM`. The `CREATEIN` privilege on the schema is granted to `PUBLIC`.

`BIND` executes under the transaction that the user has started. After performing the bind, `BIND` issues a `COMMIT` (if bind is successful) or a `ROLLBACK` (if bind is unsuccessful) operation to terminate the current transaction and start another one.

Binding halts if a fatal error or more than 100 errors occur. If a fatal error occurs during binding, `BIND` stops binding, attempts to close all files, and discards the package.

Binding application programs has prerequisite requirements and restrictions beyond the scope of this manual. For more detailed information about binding application programs to databases, see the *Embedded SQL Programming Guide*.

See Also

“PRECOMPILE PROGRAM” on page 305.

CATALOG APPC NODE

Adds an APPC node entry to the node directory. The Advanced Program-to-Program Communications protocol is used to access the remote node.

Authorization

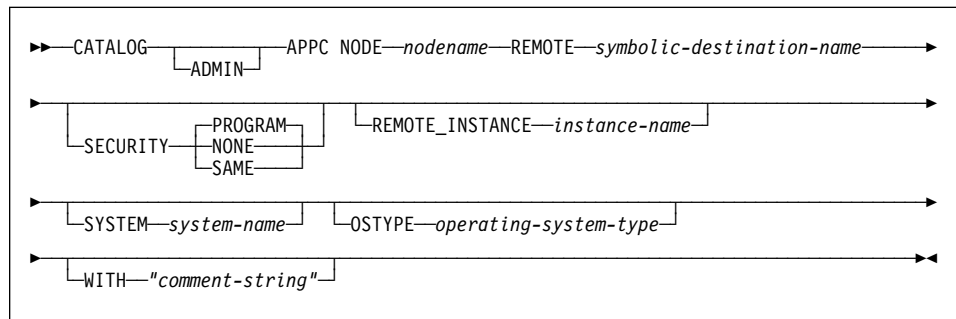
One of the following:

sysadm
sysctrl

Required Connection

None

Command Syntax



Command Parameters

ADMIN

Specifies administration server nodes.

NODE *nodename*

A local alias for the node to be cataloged. This is an arbitrary name on the user's workstation, used to identify the node. It should be a meaningful name to make it easier to remember. The name must conform to database manager naming conventions (see Appendix B, "Naming Conventions" on page 431).

REMOTE *symbolic-destination-name*

Specifies the symbolic destination name of the remote partner node. The name corresponds to an entry in the CPI Communications side information table that contains the necessary information for the client to set up an APPC connection to the server (partner LU name, mode name, partner TP name). Maximum length is 8 characters.

CATALOG APPC NODE

SECURITY

PROGRAM

Specifies that both a user name and a password are to be included in the allocation request sent to the partner LU.

NONE

Specifies that no security information is to be included in the allocation request sent to the partner LU.

SAME

Specifies that a user name is to be included in the allocation request sent to the partner LU, together with an indicator that the user name has been "already verified". The partner must be configured to accept "already verified" security.

REMOTE_INSTANCE *instance-name*

Specifies the real name of the instance to which an attachment is being made on the remote server machine.

SYSTEM *system-name*

Specifies a name that is used to identify the server machine.

OSTYPE *operating-system-type*

Specifies the operating system type of the server machine. Valid values are: OS2, AIX, WIN95, NT, HPUX, SUN, MVS, OS400, VM, VSE, SNI, SCO, and SGI.

WITH *"comment-string"*

Describes the node entry in the node directory. Any comment that helps to describe the node can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

Example

```
db2 catalog appc node db2appc1 remote db2inst1 security program
with "A remote APPC node"
```

Usage Notes

The database manager creates the node directory when the first node is cataloged (that is, when the first CATALOG...NODE command is issued). On a client using DATABASE 2 Client Application Enabler for OS/2 or DATABASE 2 Client Application Enabler for Windows, it stores and maintains the node directory in the instance subdirectory where the Client Application Enabler is installed. On a client using DATABASE 2 Client Application Enabler for AIX, it creates the node directory in the DB2 installation directory.

List the contents of the local node directory using "LIST NODE DIRECTORY" on page 258.

Note: If directory caching is enabled (see the configuration parameter *dir_cache* in "GET DATABASE MANAGER CONFIGURATION" on page 188), database, node, and DCS directory files are cached in memory. An application's directory

CATALOG APPC NODE

cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use "TERMINATE" on page 398. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

CATALOG APPCLU NODE

CATALOG APPCLU NODE

Writes information to the node directory about a remote workstation that uses APPC as its communication protocol. DB2 uses this information to establish the connection between an application and a remote database cataloged on this node.

This command is available on OS/2 only.

Authorization

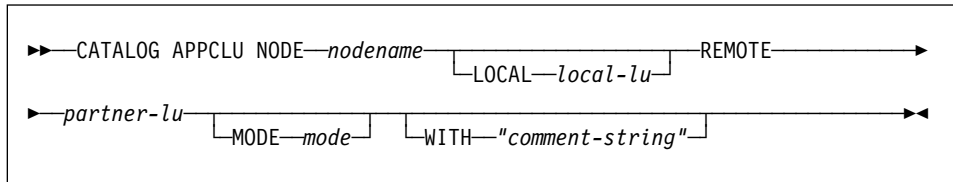
One of the following:

sysadm
sysctrl

Required Connection

None

Command Syntax



Command Parameters

NODE *nodename*

Specifies the name of the remote workstation to catalog. This is the same name that is entered for the node name parameter when cataloging a database residing on that workstation. The name must conform to DB2 naming conventions (see Appendix B, "Naming Conventions" on page 431).

LOCAL *local-lu*

Specifies the alias of the SNA local logical unit used for the connection. It must be a string containing 1 to 8 non-blank characters. The alias must be entered exactly as it appears (using mixed case characters) in the corresponding SNA definition (from the Communication Manager configuration).

REMOTE *partner-lu*

Specifies the alias of the SNA remote logical unit used for the connection. It must be a string containing 1 to 8 non-blank characters. The alias must be entered exactly as it appears (using mixed case characters) in the corresponding SNA definition (from the Communication Manager configuration).

CATALOG APPCLU NODE

MODE *mode*

Specifies the SNA transmission mode used for the connection. The name must conform to SNA naming conventions.

If a value is not entered, DB2 stores a character string of eight blanks as the mode type.

WITH *"comment-string"*

Describes the node entry in the node directory. Any comment that helps to describe the node can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

Example

The following example shows how to catalog REMNODE, a remote workstation that contains a database:

```
db2 catalog appclu node remnode local LU1 remote LU2
mode SQLMOD01 with "Remote node comment"
```

Usage Notes

This command is identical to the Version 1 CATALOG APPC NODE command, but is different from the Version 2 CATALOG APPC NODE command.

Note: Version 1 script files containing the CATALOG APPC NODE command must be modified. Change the keyword APPC to APPCLU to make this command perform as it did in Version 1.

“ATTACH” on page 91 cannot be used with an APPCLU node to attach to a server that has TPNAME other than x'07F6C4C2'.

APPCLU nodes only support security PROGRAM (security SAME and NONE are not supported).

CATALOG APPN NODE

CATALOG APPN NODE

Writes information to the node directory about a remote workstation that uses APPN as its communication protocol. DB2 uses this information to establish the connection between an application and a remote database cataloged on this node.

This command is available on OS/2 only.

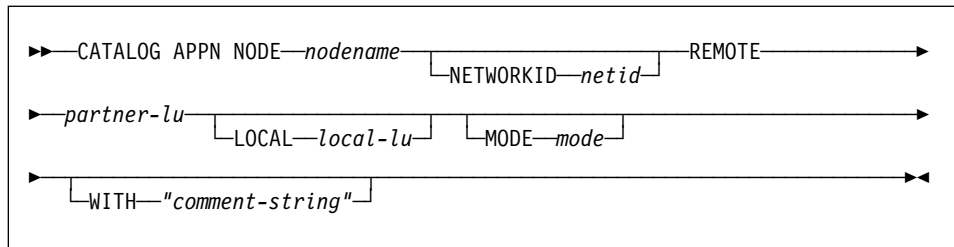
Authorization

sysadm

Required Connection

None

Command Syntax



Command Parameters

NODE *nodename*

Specifies the name of the remote workstation to catalog. This is the same name that is entered for the node name parameter when a database residing on that workstation is cataloged (using “CATALOG DATABASE” on page 118). The name must conform to DB2 naming conventions (see Appendix B, “Naming Conventions” on page 431).

NETWORKID *netid*

Specifies the ID of the SNA network where the remote LU resides. This network ID is a string of one to eight characters that follows naming conventions for SNA.

REMOTE *partner-lu*

Specifies the SNA partner logical unit used for the connection. Enter the LU name of the remote node. The name must be entered exactly as it appears (using mixed case characters) in the corresponding SNA definition (from the Communication Manager configuration). The name must follow SNA naming conventions.

LOCAL *local-lu*

Specifies the alias of the SNA local logical unit used for the connection. It must be a string containing 1 to 8 non-blank characters. The alias must be entered exactly as it appears (using mixed case characters) in the

CATALOG APPN NODE

corresponding SNA definition (from the Communication Manager configuration).

MODE *mode*

Specifies the SNA transmission mode used for the connection. The name must conform to SNA naming conventions.

If a value is not entered, DB2 stores a character string of eight blanks as the mode type.

WITH *"comment-string"*

Describes the node entry in the node directory. Any comment that helps to describe the node can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

Example

The following example catalogs an APPN node:

```
db2 catalog appn node remnode remote rlu with "Catalog APPN NODE"
```

Usage Notes

This command is identical to the Version 1 command. (It was removed in Version 2.)

“ATTACH” on page 91 cannot be used with an APPN node to attach to a server that has TPNAME other than x'07F6C4C2'.

APPN nodes only support security PROGRAM (security SAME and NONE are not supported).

CATALOG DATABASE

CATALOG DATABASE

Stores database location information in the system database directory. The database can be located either on the local workstation or on a remote node.

Scope

In a partitioned database environment, when cataloging a local database into the system database directory, this command must be issued from a node on the server where the database resides.

Authorization

One of the following:

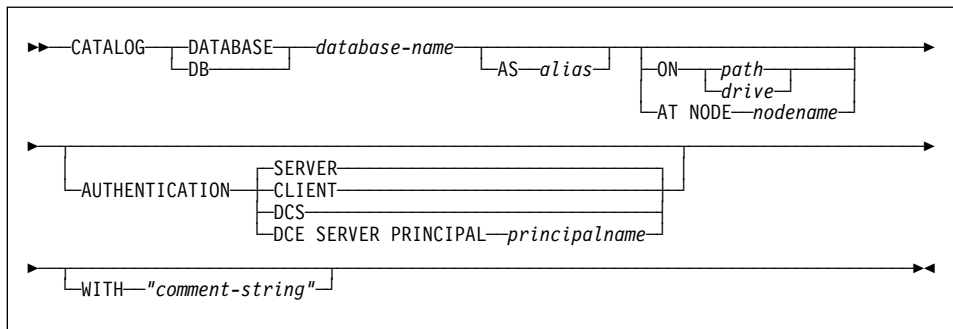
sysadm

sysctrl

Required Connection

None. Directory operations affect the local directory only.

Command Syntax



Command Parameters

DATABASE *database-name*

Specifies the name of the database to catalog.

AS *alias*

Specifies an alias as an alternate name for the database being cataloged. If an alias is not specified, the database manager uses *database-name* as the alias.

ON *path/drive*

On UNIX based systems, specifies the path on which the database being cataloged resides. On OS/2 or the Windows operating system, specifies the letter of the drive on which the database being cataloged resides.

AT NODE *nodename*

Specifies the name of the node where the database being cataloged resides. This name should match the name of an entry in the node

CATALOG DATABASE

directory. If the node name specified does not exist in the node directory, a warning is returned, but the database is cataloged in the system database directory. The node name should be cataloged in the node directory if a connection to the cataloged database is desired.

AUTHENTICATION

Note: Required only if a DB2 Version 2 or greater client is communicating with a DB2 Version 1 server.

The authentication value is stored for remote databases (it appears in the output from "LIST DATABASE DIRECTORY" on page 240) but it is not stored for local databases.

Specifying an authentication type can result in a performance benefit. For more information about authentication types, see the *Administration Guide*.

SERVER

Specifies that authentication takes place on the node containing the target database.

CLIENT

Specifies that authentication takes place on the node where the application is invoked.

DCS

Specifies that authentication takes place on the node containing the target database, except when using DB2 Connect, when it specifies that authentication takes place at the DRDA AS.

DCE

Specifies that authentication takes place using DCE Security Services. When authentication is DCE, and an APPC connection is used for access, only SECURITY=NONE is supported.

SERVER PRINCIPAL principalname

Fully qualified DCE principal name for the target server. This value is also recorded in the keytab file at the target server.

WITH *"comment-string"*

Describes the database or the database entry in the system database directory. The maximum length of a comment string is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by double quotation marks.

Example

```
db2 catalog database sample on /databases/sample  
with "Sample Database"
```

Usage Notes

Use CATALOG DATABASE to catalog databases located on local or remote nodes, recatalog databases that were uncataloged previously, or maintain multiple aliases for one database (regardless of database location).

CATALOG DATABASE

DB2 automatically catalogs databases when they are created. It catalogs an entry for the database in the local database directory and another entry in the system database directory. If the database is created from a remote client (or a client which is executing from a different instance on the same machine), an entry is also made in the system database directory at the client instance.

If neither path nor node name is specified, the database is assumed to be local, and the location of the database is assumed to be that specified in the database manager configuration parameter *dftdbpath*.

Databases on the same node as the database manager instance are cataloged as *indirect* entries. Databases on other nodes are cataloged as *remote* entries.

CATALOG DATABASE automatically creates a system database directory if one does not exist. The system database directory is stored on the path that contains the database manager instance that is being used, and is maintained outside of the database.

List the contents of the system database directory using “LIST DATABASE DIRECTORY” on page 240.

Note: If directory caching is enabled (see the configuration parameter *dir_cache* in “GET DATABASE MANAGER CONFIGURATION” on page 188), database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use “TERMINATE” on page 398. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

See Also

“UNCATALOG DATABASE” on page 399.

CATALOG DCS DATABASE

Stores information about remote databases in the Database Connection Services (DCS) directory. These databases are accessed through an Application Requester (AR), such as DB2 Connect. Having a DCS directory entry with a database name matching a database name in the system database directory invokes the specified AR to forward SQL requests to the remote server where the database resides. For more information about DB2 Connect and DCS directory entries, see the *DB2 Connect User's Guide*.

Authorization

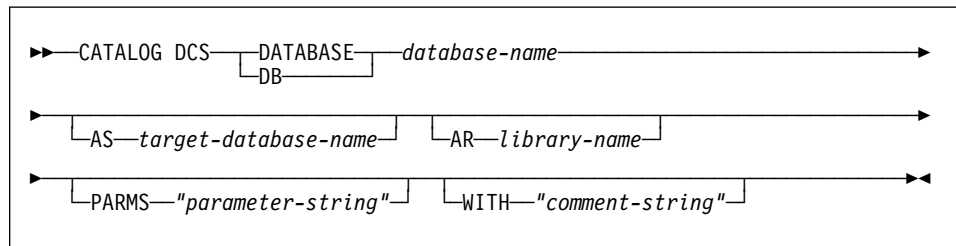
One of the following:

sysadm
sysctrl

Required Connection

None

Command Syntax



Command Parameters

DATABASE *database-name*

Specifies the alias of the target database to catalog. This alias must match the database name entered during the remote cataloging of this database in the system database directory.

AS *target-database-name*

Specifies the name of the target host database to catalog.

AR *library-name*

Specifies the name of the Application Requester library that is loaded and used to access a remote database listed in the DCS directory.

Note: If using the DB2 Connect AR, do not specify a library name. The default value will cause DB2 Connect to be invoked.

If not using DB2 Connect, specify the library name of the AR, and place that library on the same path as the database manager libraries. On OS/2 or the Windows operating system, the path is `drive:\sql11b\d11`. On UNIX based systems, the path is `$HOME/sql11b/lib` of the instance owner.

CATALOG DCS DATABASE

PARMS *"parameter-string"*

Specifies a parameter string that is to be passed to the AR when it is invoked. The parameter string must be enclosed by double quotation marks.

WITH *"comment-string"*

Describes the DCS directory entry. Any comment that helps to describe the database cataloged in this directory can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by double quotation marks.

Example

The following example catalogs information about the DB1 database, which is a DB2 for MVS host database, into the DCS directory:

```
db2 catalog dcs database db1 as dsn_db_1
with "DB2/MVS location name DSN_DB_1"
```

Usage Notes

The DB2 Connect program provides connections to DRDA Application Servers such as:

- DB2 for OS/390 databases on System/370 and System/390 architecture host computers
- DB2 for VM and VSE databases on System/370 and System/390 architecture host computers
- OS/400 databases on Application System/400 (AS/400) host computers.

The database manager creates a Database Connection Services directory if one does not exist. This directory is stored on the path that contains the database manager instance that is being used. The DCS directory is maintained outside of the database.

The database must also be cataloged as a remote database in the system database directory.

List the contents of the DCS directory using "LIST DCS DIRECTORY" on page 247.

Note: If directory caching is enabled (see the configuration parameter *dir_cache* in "GET DATABASE MANAGER CONFIGURATION" on page 188), database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use "TERMINATE" on page 398. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

See Also

“UNCATALOG DCS DATABASE” on page 401.

CATALOG GLOBAL DATABASE

CATALOG GLOBAL DATABASE

Creates a system database directory entry of the DCE type. This entry is used to define a local alias of the fully qualified DCE directory object name of the target database. The information about that database is stored centrally in the DCE directory.

Authorization

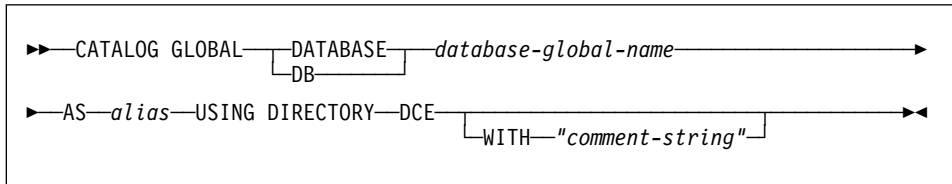
One of the following:

sysadm
sysctrl

Required Connection

None

Command Syntax



Command Parameters

DATABASE *database-global-name*

Specifies the fully qualified name that uniquely identifies the database in the DCE name space.

AS *alias*

Specifies an alternate name for the database being cataloged.

USING DIRECTORY *DCE*

Specifies the global directory service being used.

WITH *"comment-string"*

Describes the DCE type entry in the system database directory. Any comment that helps to describe the database cataloged in this directory can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by double quotation marks.

Example

```
db2 catalog global database ../../cell1/subsys/database/DB3  
as dbtest using directory dce
```


Usage Notes

The maximum length of *database-global-name* is 255 bytes. The name must begin with either */.../* or *././*.

Note: If directory caching is enabled (see the configuration parameter *dir_cache* in “GET DATABASE MANAGER CONFIGURATION” on page 188), database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use “TERMINATE” on page 398. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

CATALOG IPX/SPX NODE

CATALOG IPX/SPX NODE

Adds an Internetwork Packet Exchange/Sequenced Packet Exchange (IPX/SPX) node entry to the node directory. The Novell NetWare IPX/SPX communications protocol is used to access the remote node.

This command is available on OS/2, Windows NT, and Windows 95 only.

Scope

IPX/SPX file server addressing is not supported in a multi-node environment.

Authorization

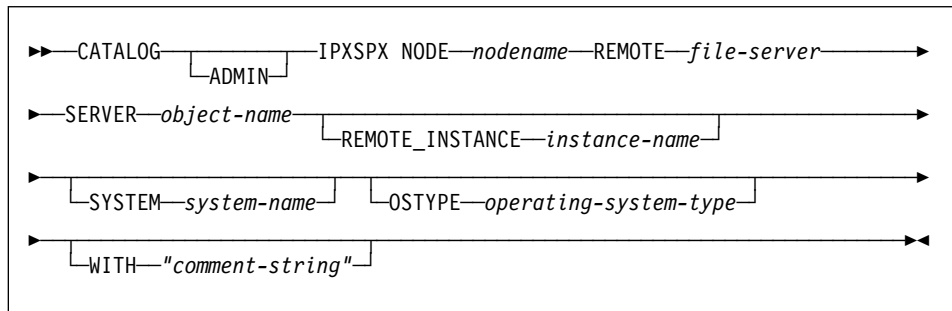
One of the following:

sysadm
sysctrl

Required Connection

None

Command Syntax



Command Parameters

ADMIN

Specifies that an IPX/SPX administration server node is to be cataloged.

NODE *nodename*

A local alias for the node to be cataloged. This is an arbitrary name on the user's workstation, used to identify the node. It should be a meaningful name to make it easier to remember. The name must conform to database manager naming conventions (see Appendix B, "Naming Conventions" on page 431).

REMOTE *file-server*

Name of the NetWare file server where the internetwork address of the server database manager instance is registered. The internetwork address

CATALOG IPX/SPX NODE

is stored in the bindery at the NetWare file server, and is accessed using *object-name*.

Note: The following characters are not valid: / \ ; , * ?

SERVER *object-name*

Name of the database manager instance stored in the bindery of the NetWare file server. Each server database manager instance registered at one NetWare file server must be represented by a unique *object-name*. It is recommended that each database manager instance on the network be represented by a unique *object-name*.

Note: The following characters are not valid: / \ ; , * ?

When cataloging the IPX/SPX client to use *file server* addressing, specify the file server and object name as defined above. When cataloging the IPX/SPX client to use *direct* addressing, specify *file-server* as *, and specify the server's IPX/SPX internetwork address in the *object-name* parameter. Use "db2ipxad - Get IPX/SPX Internetwork Address" on page 36 to retrieve the server's IPX/SPX internetwork address. For more information about the addressing methods, see one of the *Quick Beginnings* books.

REMOTE_INSTANCE *instance-name*

Specifies the name of the server instance to which an attachment is being made.

SYSTEM *system-name*

Specifies the DB2 system name that is used to identify the server machine.

OSTYPE *operating-system-type*

Specifies the operating system type of the server machine. Valid values are: OS2, AIX, WIN95, NT, HPUX, SUN, MVS, OS400, VM, VSE, SNI, SCO, and SGI.

WITH *"comment-string"*

Describes the node entry in the node directory. Any comment that helps to describe the node can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

Examples

```
db2 catalog ipxspx node db2ipx1 remote netwsrv server db2inst1
with "A remote IPX/SPX node"
```

```
db2 catalog ipxspx node db2ipx2 remote * server 09212700.400011527745.879E
with "IPX/SPX node using direct addr"
```

Usage Notes

The database manager creates the node directory when the first node is cataloged (that is, when the first CATALOG...NODE command is issued). On a client using DATABASE 2 Client Application Enabler for OS/2 or DATABASE 2 Client Application Enabler for Windows, it stores and maintains the node directory in the instance subdirectory where

CATALOG IPX/SPX NODE

the Client Application Enabler is installed. On a client using DATABASE 2 Client Application Enabler for AIX, it creates the node directory in the DB2 installation directory.

List the contents of the local node directory using “LIST NODE DIRECTORY” on page 258.

Note: If directory caching is enabled (see the configuration parameter *dir_cache* in “GET DATABASE MANAGER CONFIGURATION” on page 188), database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use “TERMINATE” on page 398. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

CATALOG LOCAL NODE

Creates a local alias for an instance that resides on the same machine. A local node should be cataloged when there is more than one instance on the same workstation to be accessed from the user's client. Interprocess Communications (IPC) is used to access the local node.

Authorization

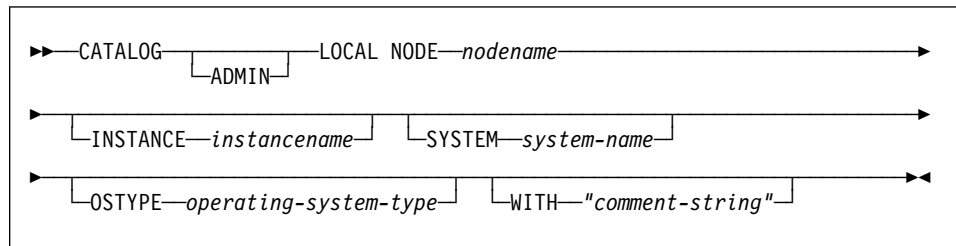
One of the following:

sysadm
sysctrl

Required Connection

None

Command Syntax



Command Parameters

ADMIN

Specifies that a local administration server node is to be cataloged.

NODE *nodename*

A local alias for the node to be cataloged. This is an arbitrary name on the user's workstation, used to identify the node. It should be a meaningful name to make it easier to remember. The name must conform to database manager naming conventions (see Appendix B, "Naming Conventions" on page 431).

INSTANCE *instancename*

Name of the local instance to be accessed.

SYSTEM *system-name*

Specifies the DB2 system name that is used to identify the server machine.

OSTYPE *operating-system-type*

Specifies the operating system type of the server machine. Valid values are: OS2, AIX, WIN95, NT, HPUX, SUN, MVS, OS400, VM, VSE, SNI, SCO, and SGI.

CATALOG LOCAL NODE

WITH *"comment-string"*

Describes the node entry in the node directory. Any comment that helps to describe the node can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

Example

Workstation A has two server instances, `inst1` and `inst2`. To create databases at both instances from a single CLP session, issue the following sequence of commands (assume the **DB2INSTANCE** environment variable is set to `inst1`):

1. Create a local database at `inst1`:

```
db2 create database mydb1
```
2. Catalog another server instance on this workstation:

```
db2 catalog local node mynode2 instance inst2
```
3. Create a database at `mynode2`:

```
db2 attach to mynode2  
db2 create database mydb2
```

Usage Notes

Note: If directory caching is enabled (see the configuration parameter `dir_cache` in “GET DATABASE MANAGER CONFIGURATION” on page 188), database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use “TERMINATE” on page 398. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

CATALOG NAMED PIPE NODE

Adds a named pipe node entry to the node directory. The named pipe is used to access the remote node.

This command is available on Windows NT only.

Authorization

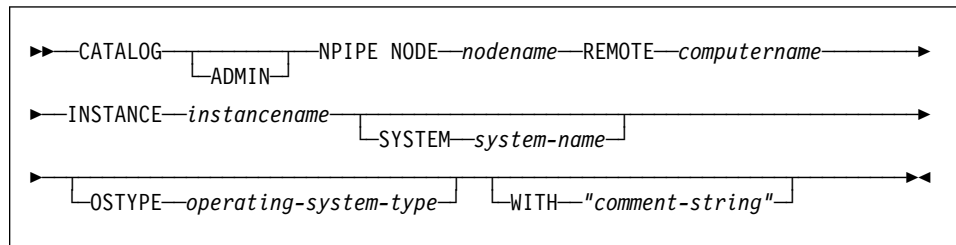
One of the following:

sysadm
sysctrl

Required Connection

None

Command Syntax



Command Parameters

ADMIN

Specifies that an NPIPE administration server node is to be cataloged.

NODE *nodename*

A local alias for the node to be cataloged. This is an arbitrary name on the user's workstation, used to identify the node. It should be a meaningful name to make it easier to remember. The name must conform to database manager naming conventions (see Appendix B, "Naming Conventions" on page 431).

REMOTE *computername*

The computer name of the node on which the target database resides. Maximum length is 15 characters.

INSTANCE *instancename*

Name of the server instance on which the target database resides. Identical to the name of the remote named pipe, which is used to communicate with the remote node.

CATALOG NAMED PIPE NODE

SYSTEM *system-name*

Specifies the DB2 system name that is used to identify the server machine.

OSTYPE *operating-system-type*

Specifies the operating system type of the server machine. Valid values are: OS2, AIX, WIN95, NT, HPUX, SUN, MVS, OS400, VM, VSE, SNI, SCO, and SGI.

WITH *"comment-string"*

Describes the node entry in the node directory. Any comment that helps to describe the node can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

Example

```
db2 catalog npipe node db2np1 remote nphost instance db2inst1
with "A remote named pipe node."
```

Usage Notes

The database manager creates the node directory when the first node is cataloged (that is, when the first CATALOG...NODE command is issued). On a client using DATABASE 2 Client Application Enabler for OS/2 or DATABASE 2 Client Application Enabler for Windows, it stores and maintains the node directory in the instance subdirectory where the Client Application Enabler is installed. On a client using DATABASE 2 Client Application Enabler for AIX, it creates the node directory in the DB2 installation directory.

List the contents of the local node directory using "LIST NODE DIRECTORY" on page 258.

Note: If directory caching is enabled (see the configuration parameter *dir_cache* in "GET DATABASE MANAGER CONFIGURATION" on page 188), database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use "TERMINATE" on page 398. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

CATALOG NETBIOS NODE

Adds a NetBIOS node entry to the node directory. The NetBIOS communications protocol is used to access the remote node.

This command is available on OS/2, Windows NT, and Windows 95 only.

Authorization

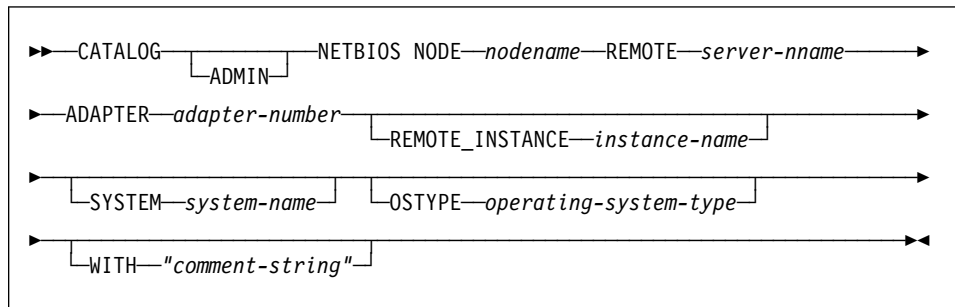
One of the following:

sysadm
sysctrl

Required Connection

None. Directory operations affect the local directory only.

Command Syntax



Command Parameters

ADMIN

Specifies administration server nodes.

NODE *nodename*

A local alias for the node to be cataloged. This is an arbitrary name on the user's workstation, used to identify the node. It should be a meaningful name to make it easier to remember. The name must conform to database manager naming conventions (see Appendix B, "Naming Conventions" on page 431).

REMOTE *server-nname*

The name of the remote workstation where the target database resides. This name must conform to the naming conventions for the database manager. This is the workstation name (*nname*) found in the database manager configuration file of the server workstation.

CATALOG NETBIOS NODE

ADAPTER *adapter-number*

Specifies the local, logical, outgoing LAN adapter number. The default value is zero.

REMOTE_INSTANCE *instance-name*

Specifies the real name of the instance to which an attachment is being made on the remote server machine.

SYSTEM *system-name*

Specifies a name that is used to identify the server machine.

OSTYPE *operating-system-type*

Specifies the operating system type of the server machine. Valid values are: OS2, AIX, WIN95, NT, HPUX, SUN, MVS, OS400, VM, VSE, SNI, SCO, and SGI.

WITH *"comment-string"*

Describes the node entry in the node directory. Any comment that helps to describe the node can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

Example

```
db2 catalog netbios node db2netb1 remote db2inst1 adapter 0
with "A remote NetBIOS node"
```

Usage Notes

The database manager creates the node directory when the first node is cataloged (that is, when the first CATALOG...NODE command is issued). On a client using DATABASE 2 Client Application Enabler for OS/2 or DATABASE 2 Client Application Enabler for Windows, it stores and maintains the node directory in the instance subdirectory where the Client Application Enabler is installed. On a client using DATABASE 2 Client Application Enabler for AIX, it creates the node directory in the DB2 installation directory.

List the contents of the local node directory using "LIST NODE DIRECTORY" on page 258.

Note: If directory caching is enabled (see the configuration parameter *dir_cache* in "GET DATABASE MANAGER CONFIGURATION" on page 188), database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use "TERMINATE" on page 398. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

CATALOG ODBC DATA SOURCE

Catalogs a user or system ODBC data source.

A *data source*, in ODBC (Open Database Connectivity) terminology, is a user-defined name for a specific database or file system. That name is used to access the database or file system through ODBC APIs. On Windows NT and Windows 95, either user or system data sources can be cataloged. A user data source is only visible to the user who cataloged it, whereas a system data source is visible to and can be used by all other users.

This command is available on Windows NT, Windows 95, and Windows 3.1 only.

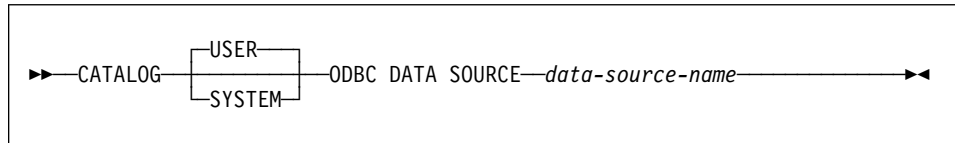
Authorization

None

Required Connection

None

Command Syntax



Command Parameters

USER

Catalog a user data source. This is the default if no keyword is specified.

SYSTEM

Catalog a system data source.

ODBC DATA SOURCE *data-source-name*

Specifies the name of the data source to be cataloged. Maximum length is 32 characters.

See Also

“LIST ODBC DATA SOURCES” on page 264

“UNCATALOG ODBC DATA SOURCE” on page 404.

CATALOG TCP/IP NODE

CATALOG TCP/IP NODE

Adds a Transmission Control Protocol/Internet Protocol (TCP/IP) node entry to the node directory. The TCP/IP communications protocol is used to access the remote node.

Authorization

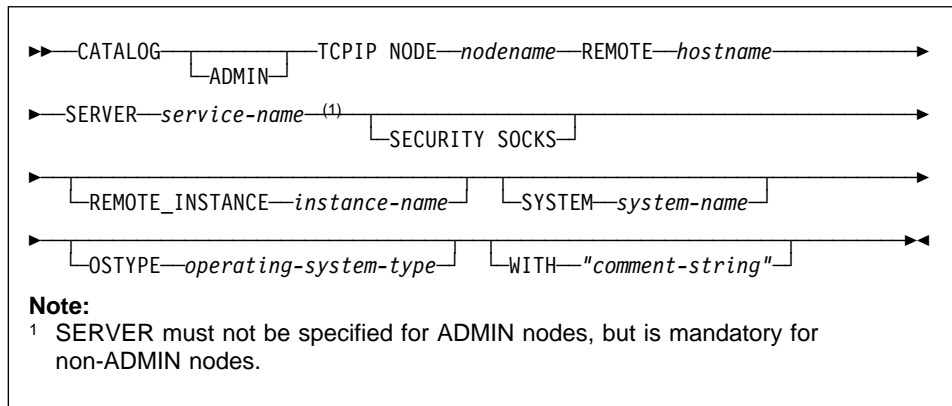
One of the following:

sysadm
sysctrl

Required Connection

None. Directory operations affect the local directory only.

Command Syntax



Command Parameters

ADMIN

Specifies that a TCP/IP administration server node is to be cataloged.

NODE *nodename*

A local alias for the node to be cataloged. This is an arbitrary name on the user's workstation, used to identify the node. It should be a meaningful name to make it easier to remember. The name must conform to database manager naming conventions (see Appendix B, "Naming Conventions" on page 431).

REMOTE *hostname*

The host name of the node where the target database resides. The host name is the name of the node that is known to the TCP/IP network. Maximum length is 255 characters.

CATALOG TCP/IP NODE

SERVER *service-name*

Specifies the service name or the port number of the server database manager instance.

The CATALOG TCPIP NODE command is run on a client.

- If a service name is specified, the *services* file on the client is used to map the service name to a port number. A service name is specified in the database manager configuration file, and the *services* file on the server is used to map this service name to a port number. The port number on the client and the server must match.

Note: A port number, instead of a service name, can be specified in the database manager configuration file on the server, but this is not recommended.

- If a port number is specified, it must match the port number associated with the service name specified in the server's database manager configuration file. No service name needs to be specified in the local TCP/IP *services* file.

The value of *service-name* is used as a key to search the local *services* file for the associated port number. If a matching entry is not found, and *service-name* is numeric, the value is interpreted as the port number.

Maximum length is 14 characters. This parameter is case sensitive.

Note: This parameter must not be specified for ADMIN nodes. The value on ADMIN nodes is always 523.

SECURITY SOCKS

Specifies that the node will be SOCKS-enabled.

The following environment variables are mandatory and *must* be set to enable SOCKS:

SOCKS_NS

The Domain Name Server for resolving the host address of the SOCKS server. This should be an IP address.

SOCKS_SERVER

The fully qualified host name or the IP address of the SOCKS server. If the SOCKSified DB2 client is unable to resolve the fully qualified host name, it assumes that an IP address has been entered.

One of the following conditions should be true:

- The SOCKS server should be reachable via the domain name server
- It should be listed in the hosts file. The location of this file is described in the TCP/IP documentation.
- It should be in an IP address format.

If these environment variables are set after a **db2start** has been issued, it is necessary to issue a TERMINATE command.

CATALOG TCP/IP NODE

REMOTE_INSTANCE *instance-name*

Specifies the name of the server instance to which an attachment is being made.

SYSTEM *system-name*

Specifies the DB2 system name that is used to identify the server machine.

OSTYPE *operating-system-type*

Specifies the operating system type of the server machine. Valid values are: OS2, AIX, WIN95, NT, HPUX, SUN, MVS, OS400, VM, VSE, SNI, SCO, and SGI.

WITH *"comment-string"*

Describes the node entry in the node directory. Any comment that helps to describe the node can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

Examples

```
db2 catalog tcpip node db2tcp1 remote tcp host server db2inst1
with "A remote TCP/IP node"
```

```
db2 catalog tcpip node db2tcp2 remote 9.21.15.235 server db2inst2
with "TCP/IP node using IP address"
```

Usage Notes

The database manager creates the node directory when the first node is cataloged (that is, when the first CATALOG...NODE command is issued). On a client using DATABASE 2 Client Application Enabler for OS/2 or DATABASE 2 Client Application Enabler for Windows, it stores and maintains the node directory in the instance subdirectory where the Client Application Enabler is installed. On a client using DATABASE 2 Client Application Enabler for AIX, it creates the node directory in the DB2 installation directory.

List the contents of the local node directory using "LIST NODE DIRECTORY" on page 258.

Note: If directory caching is enabled (see the configuration parameter *dir_cache* in "GET DATABASE MANAGER CONFIGURATION" on page 188), database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use "TERMINATE" on page 398. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

CHANGE DATABASE COMMENT

Changes a database comment in the system database directory or the local database directory. New comment text can be substituted for text currently associated with a comment.

Scope

This command only affects the node on which it is executed.

Authorization

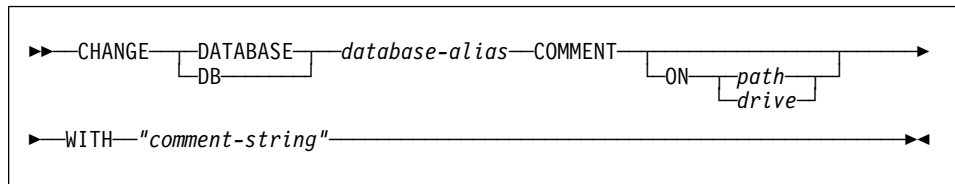
One of the following:

sysadm
sysctrl

Required Connection

None

Command Syntax



Command Parameters

DATABASE *database-alias*

Specifies the alias of the database whose comment is to be changed. To change the comment in the system database directory, specify the alias for the database. To change the comment in the local database directory, specify the path where the database resides (with the *path* parameter), and enter the name (not the alias) of the database.

ON *path/drive*

On UNIX based systems, specifies the path on which the database resides, and changes the comment in the local database directory. If a path is not specified, the database comment for the entry in the system database directory is changed. On OS/2 or the Windows operating system, specifies the letter of the drive on which the database resides.

WITH *"comment-string"*

Describes the entry in the system database directory or the local database directory. Any comment that helps to describe the cataloged database can be entered. The maximum length of a comment string is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by double quotation marks.

CHANGE DATABASE COMMENT

Example

The following example changes the text in the system database directory comment for the SAMPLE database from "Test 2 - Holding" to "Test 2 - Add employee inf rows":

```
db2 change database sample comment  
with "Test 2 - Add employee inf rows"
```

Usage Notes

New comment text replaces existing text. To append information, enter the old comment text, followed by the new text.

Only the comment for an entry associated with the database alias is modified. Other entries with the same database name, but with different aliases, are not affected.

If the path is specified, the database alias must be cataloged in the local database directory. If the path is not specified, the database alias must be cataloged in the system database directory.

See Also

"CREATE DATABASE" on page 143.

CHANGE ISOLATION LEVEL

CHANGE ISOLATION LEVEL

Changes the way that DB2 isolates data from other processes while a database is being accessed.

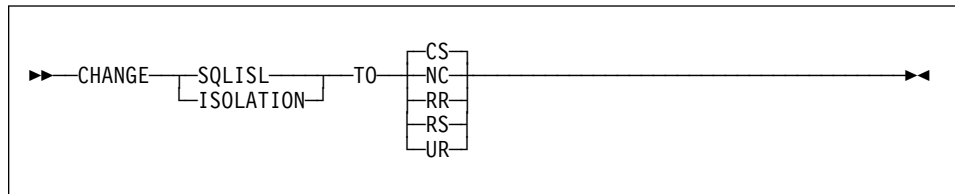
Authorization

None

Required Connection

None

Command Syntax



Command Parameters

TO

CS

Specifies cursor stability as the isolation level.

NC

Specifies no commit as the isolation level. Not supported by DB2.

RR

Specifies repeatable read as the isolation level.

RS

Specifies read stability as the isolation level.

UR

Specifies uncommitted read as the isolation level.

Usage Notes

DB2 uses isolation levels to maintain data integrity in a database. The isolation level defines the degree to which an application process is isolated (shielded) from changes made by other concurrently executing application processes.

If a selected isolation level is not supported by a database, it is automatically escalated to a supported level at connect time.

Isolation level changes are not permitted while connected to a database with a type 1 connection (see "SET CLIENT" on page 382). Changes are permitted using a type 2 connection, but should be made with caution, because the changes will apply to every connection made from the same command line processor back-end process. The user

CHANGE ISOLATION LEVEL

assumes responsibility for remembering which isolation level applies to which connected database.

In the following example, a user is in DB2 interactive mode following creation of the SAMPLE database:

```
update command options using c off
catalog db sample as sample2

set client connect 2

connect to sample
connect to sample2

change isolation to cs
set connection sample
declare c1 cursor for select * from org
open c1
fetch c1 for 3 rows

change isolation to rr
fetch c1 for 2 rows
```

An SQL0514N error occurs because c1 is not in a prepared state for this isolation level.

```
change isolation to cs
set connection sample2
fetch c1 for 2 rows
```

An SQL0514N error occurs because c1 is not in a prepared state for this database.

```
declare c1 cursor for select division from org
```

A DB21029E error occurs because cursor c1 has already been declared and opened.

```
set connection sample
fetch c1 for 2 rows
```

This works because the original database (SAMPLE) was used with the original isolation level (CS).

For more information about isolation levels, see the *SQL Reference*.

See Also

“QUERY CLIENT” on page 326.

CREATE DATABASE

Initializes a new database with an optional user-defined collating sequence, creates the three initial table spaces, creates the system tables, and allocates the recovery log.

This command is not valid on a client.

Scope

In a multi-node environment, this command affects all nodes that are listed in the `db2nodes.cfg` file.

The node from which this command is issued becomes the catalog node for the new database.

Authorization

One of the following:

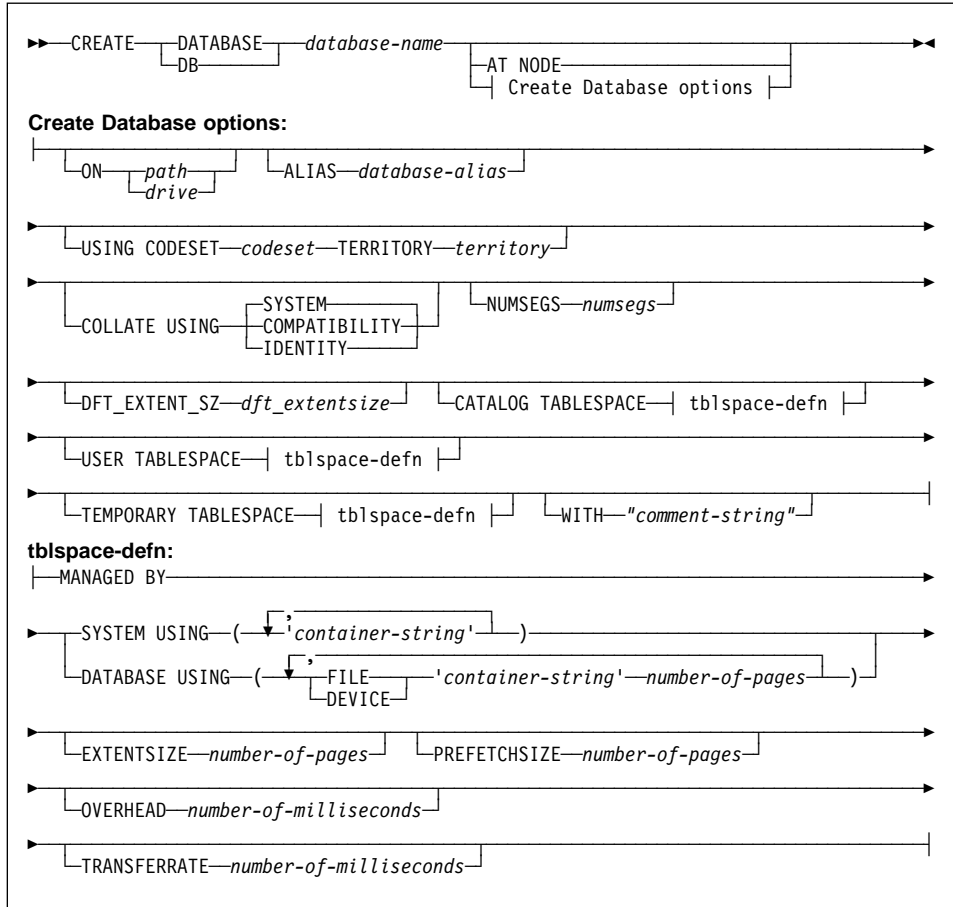
sysadm
sysctrl

Required Connection

Instance. To create a database at another (remote) node, it is necessary to first attach to that node. A database connection is temporarily established by this command during processing.

CREATE DATABASE

Command Syntax



Notes:

1. The code set and territory values specified must be a valid combination. For a list of valid combinations, see one of the *Quick Beginnings* books.
2. For details on the **tblspace-defn** parameters, see the CREATE TABLESPACE statement in the *SQL Reference*. The table space definitions specified on CREATE DATABASE apply to all nodes on which the database is being created. They cannot be specified separately for each node. If the table space definitions are to be created differently on particular nodes, the CREATE TABLESPACE statement must be used.

Command Parameters

DATABASE *database-name*

A name to be assigned to the new database. This must be a unique name that differentiates the database from any other database in either the local

CREATE DATABASE

database directory or the system database directory. The name must conform to naming conventions for databases.

AT NODE

Specifies that the database is to be created only on the node that issues the command. This parameter is not intended for general use. For example, it should be used with “RESTORE DATABASE” on page 362 if the database partition at a node was damaged and must be re-created. Improper use of this parameter can cause inconsistencies in the system, so it should only be used with caution.

Note: If this parameter is used to recreate a database partition that was dropped (because it was damaged), the database at this node will be in the restore-pending state. After recreating the database partition, the database must immediately be restored on this node.

ON *path/drive*

On UNIX based systems, specifies the path on which to create the database. If a path is not specified, the database is created on the default database path specified in the database manager configuration file (*dftdbpath* parameter). Maximum length is 205 characters. On OS/2 or the Windows operating system, specifies the letter of the drive on which to create the database.

Note: For MPP systems, a database should not be created in an NFS-mounted directory. If a path is not specified, ensure that the *dftdbpath* database manager configuration parameter is not set to an NFS-mounted path (for example, on UNIX based systems, it should not specify the \$HOME directory of the instance owner). The path specified for this command in an MPP system cannot be a relative path.

ALIAS *database-alias*

An alias for the database in the system database directory. If no alias is provided, the specified database name is used.

USING CODESET *codeset*

Specifies the code set to be used for data entered into this database.

TERRITORY *territory*

Specifies the territory to be used for data entered into this database.

COLLATE USING

Identifies the type of collating sequence to be used for the database. Once the database has been created, the collating sequence cannot be changed.
COMPATIBILITY

The DB2 Version 2 collating sequence. Some collation tables have been enhanced. This option specifies that the previous version of these tables is to be used.

IDENTITY

Identity collating sequence, in which strings are compared byte for byte.

SYSTEM

Collating sequence based on the current territory.

CREATE DATABASE

For information about how the database collating sequence is used, see the *SQL Reference*.

NUMSEGS *numsegs*

Specifies the number of segment directories that will be created and used to store DAT, IDX, LF, LB, and LBA files for any default SMS table spaces. This parameter does not affect DMS table spaces, any SMS table spaces with explicit creation characteristics (created when the database is created), or any SMS table spaces explicitly created after the database is created.

DFT_EXTENT_SZ *dft_extentsize*

Specifies the default extent size of table spaces in the database.

CATALOG TABLESPACE *tblspace-defn*

Specifies the definition of the table space which will hold the catalog tables, SYSCATSPACE. If not specified, SYSCATSPACE will be created as a System Managed Space (SMS) table space with *numsegs* number of directories as containers, and with an extent size of *dft_extentsize*. For example, the following containers would be created if *numsegs* were specified to be 5:

```
/u/smith/smith/NODE0000/SQL00001/SQLT0000.0  
/u/smith/smith/NODE0000/SQL00001/SQLT0000.1  
/u/smith/smith/NODE0000/SQL00001/SQLT0000.2  
/u/smith/smith/NODE0000/SQL00001/SQLT0000.3  
/u/smith/smith/NODE0000/SQL00001/SQLT0000.4
```

In an MPP system, the catalog table space is only created on the catalog node (the node on which the CREATE DATABASE command is issued).

USER TABLESPACE *tblspace-defn*

Specifies the definition of the initial user table space, USERSPACE1. If not specified, USERSPACE1 will be created as an SMS table space with *numsegs* number of directories as containers, and with an extent size of *dft_extentsize*. For example, the following containers would be created if *numsegs* were specified to be 5:

```
/u/smith/smith/NODE0000/SQL00001/SQLT0001.0  
/u/smith/smith/NODE0000/SQL00001/SQLT0001.1  
/u/smith/smith/NODE0000/SQL00001/SQLT0001.2  
/u/smith/smith/NODE0000/SQL00001/SQLT0001.3  
/u/smith/smith/NODE0000/SQL00001/SQLT0001.4
```

TEMPORARY TABLESPACE *tblspace-defn*

Specifies the definition of the initial temporary table space, TEMPSPACE1. If not specified, TEMPSPACE1 will be created as an SMS table space with *numsegs* number of directories as containers, and with an extent size of *dft_extentsize*. For example, the following containers would be created if *numsegs* were specified to be 5:

```
/u/smith/smith/NODE0000/SQL00001/SQLT0002.0  
/u/smith/smith/NODE0000/SQL00001/SQLT0002.1  
/u/smith/smith/NODE0000/SQL00001/SQLT0002.2  
/u/smith/smith/NODE0000/SQL00001/SQLT0002.3  
/u/smith/smith/NODE0000/SQL00001/SQLT0002.4
```

CREATE DATABASE

WITH "comment-string"

Describes the database entry in the database directory. Any comment that helps to describe the database can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

Usage Notes

CREATE DATABASE:

- Creates a database in the specified subdirectory. In an MPP system, creates the database on all nodes listed in `db2nodes.cfg`, and creates a `$DB2INSTANCE/NODExxxx` directory under the specified subdirectory at each node. In a non-MPP system, creates a `$DB2INSTANCE/NODE0000` directory under the specified subdirectory.
- Creates the system catalog tables and recovery log.
- Catalogs the database in the following database directories:
 - server's local database directory on the path indicated by *path* or, if the path is not specified, the default database path defined in the database manager system configuration file. A local database directory resides on each file system that contains a database.
 - server's system database directory for the attached instance. The resulting directory entry will contain the database name and a database alias.

If the command was issued from a remote client, the client's system database directory is also updated with the database name and an alias.

Creates a system or a local database directory if neither exists. If specified, the comment and code set values are placed in both directories.

- Stores the specified code set, territory, and collating sequence. A flag is set in the database configuration file if the collating sequence consists of unique weights, or if it is the identity sequence.
- Creates the schemata called SYSCAT, SYSFUN, SYSIBM, and SYSSTAT with SYSIBM as the owner. The server node on which this command is issued becomes the catalog node for the new database. Two nodegroups are created automatically: IBMDEFAULTGROUP and IBMCATGROUP. For more information, see the *SQL Reference*.
- Binds the previously defined database manager bind files to the database (these are listed in the utilities bind file list, `db2ubind.lst`). If one or more of these files do not bind successfully, CREATE DATABASE returns a warning in the SQLCA, and provides information about the binds that failed. If a bind fails, the user can take corrective action and manually bind the failing file. The database is created in any case. A schema called NULLID is implicitly created when performing the binds with CREATEIN privilege granted to PUBLIC.

Note: The utilities bind file list contains two bind files that cannot be bound against down-level servers:

CREATE DATABASE

- `db2ugtpi.bnd` cannot be bound against DB2 Version 2 servers.
- `db2dropv.bnd` cannot be bound against DB2 Parallel Edition Version 1 servers.

If `db2ubind.lst` is bound against a down-level server, warnings pertaining to these two files are returned, and can be disregarded.

- Creates SYSCATSPACE, TEMPSPACE1, and USERSPACE1 table spaces. The SYSCATSPACE table space is only created on the catalog node.
- Grants the following:
 - DBADM authority, and CONNECT, CREATETAB, BINDADD, CREATE_NOT_FENCED, and IMPLICIT_SCHEMA privileges to the database creator
 - CONNECT, CREATETAB, BINDADD, and IMPLICIT_SCHEMA privileges to PUBLIC
 - SELECT privilege on each system catalog to PUBLIC
 - BIND and EXECUTE privilege to PUBLIC for each successfully bound utility.

With *dbadm* authority, one can grant these privileges to (and revoke them from) other users or PUBLIC. If another administrator with *sysadm* or *dbadm* authority over the database revokes these privileges, the database creator nevertheless retains them.

In an MPP environment, the database manager creates a subdirectory, `$DB2INSTANCE/NODExxxx`, under the specified or default path on all nodes. The *xxxx* is the node number as defined in the `db2nodes.cfg` file (that is, node 0 becomes `NODE0000`). Subdirectories `SQL00001` through `SQLnnnnn` will reside on this path. This ensures that the database objects associated with different nodes are stored in different directories (even if the subdirectory `$DB2INSTANCE` under the specified or default path is shared by all nodes).

CREATE DATABASE will fail if the application is already connected to a database.

Use CATALOG DATABASE to define different alias names for the new database.

See Also

- “BIND” on page 98
- “CATALOG DATABASE” on page 118
- “DROP DATABASE” on page 157.

DEACTIVATE DATABASE

Stops the specified database.

Scope

In an MPP system, this command deactivates the specified database on all nodes in the system. If one or more of these nodes encounters an error, a warning is returned. The database will be successfully deactivated on some nodes, but may remain activated on the nodes encountering the error.

Authorization

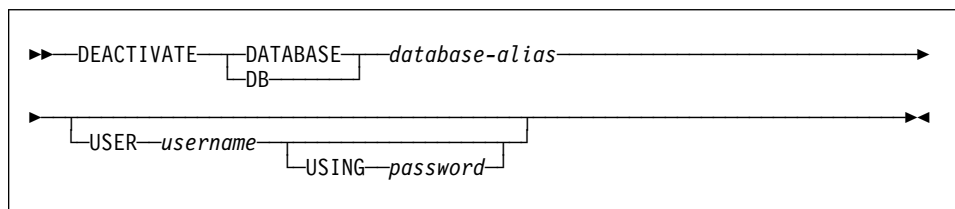
One of the following:

- sysadm*
- sysctrl*
- sysmaint*

Required Connection

None

Command Syntax



Command Parameters

DATABASE *database-alias*
Specifies the alias of the database to be stopped.

USER *username*
Specifies the user stopping the database.

USING *password*
Specifies the password for the user ID.

Usage Notes

Databases initialized by **ACTIVATE DATABASE** can be shut down by **DEACTIVATE DATABASE** or by **db2stop**. If a database was initialized by **ACTIVATE DATABASE**, the last application disconnecting from the database will not shut down the database, and **DEACTIVATE DATABASE** must be used. (In this case, **db2stop** will also shut down the database.)

Note: The application issuing the **DEACTIVATE DATABASE** command cannot have an active database connection to any database.

DEACTIVATE DATABASE

See Also

“ACTIVATE DATABASE” on page 87
“STOP DATABASE MANAGER” on page 395.

DEREGISTER

Deregisters the DB2 server from the network server. The DB2 server's network address is removed from the network server.

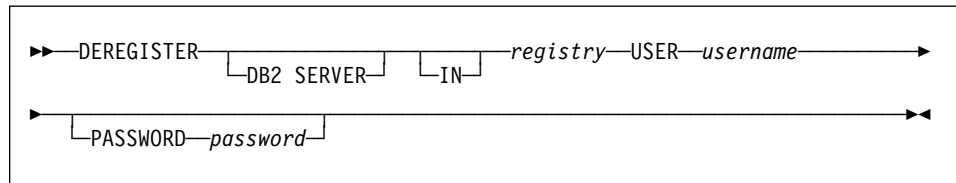
This command is not available on the Windows operating system.

Authorization

None

Required Connection

None

Command Syntax**Command Parameters**

IN *registry*

Indicates where on the network server to deregister the DB2 server. In this release, the only supported value is NWBINDERY (NetWare bindery).

USER *username*

User ID to log into the network server. The user ID must have SUPERVISOR or Workgroup Manager security equivalence.

PASSWORD *password*

Password used to log into the network server. The password must have SUPERVISOR or Workgroup Manager security equivalence.

Usage Notes

This command *must* be issued locally from the DB2 server. It is not supported remotely.

This command is only relevant when using file server addressing to connect a client and server.

See Also

“REGISTER” on page 340.

DESCRIBE

DESCRIBE

This command:

- Displays the SQLDA information about a SELECT statement
- Displays columns of a table or a view
- Displays indexes of a table or a view

Authorization

To display the SQLDA information about a SELECT statement, one of the privileges or authorities listed below for each table or view referenced in the SELECT statement is required.

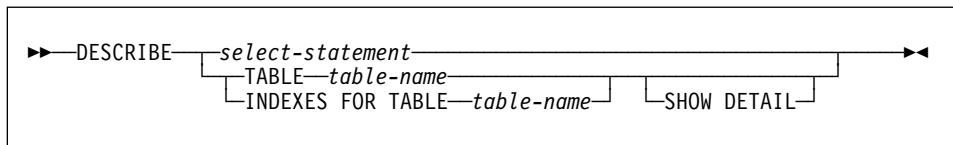
To display the columns or indexes of a table or a view, one of the privileges or authorities listed below for the system catalogs SYSCAT.COLUMNS (DESCRIBE TABLE) and SYSCAT.INDEXES (DESCRIBE INDEXES FOR TABLE) is required:

SELECT privilege
CONTROL privilege
sysadm or *dbadm* authority

Required Connection

Database. If implicit connect is enabled, a connection to the default database is established.

Command Syntax



Command Parameters

select-statement

Identifies the statement about which information is wanted. The SELECT statement is automatically prepared by CLP.

TABLE *table-name*

Specifies the table or view to be described. The fully qualified name or alias in the form *schema.table-name* must be used. The *schema* is the user name under which the table or view was created.

The DESCRIBE TABLE command lists the following information about each column:

- Column name
- Type schema
- Type name

DESCRIBE

- Length
- Scale
- Nulls (yes/no)

INDEXES FOR TABLE *table-name*

Specifies the table or view for which indexes need to be described. The fully qualified name or alias in the form *schema.table-name* must be used. The *schema* is the user name under which the table or view was created.

The DESCRIBE INDEXES FOR TABLE command lists the following information about each index of the table or view:

- Index schema
- Index name
- Unique rule
- Column count

SHOW DETAIL

For the DESCRIBE TABLE command, specifies that output include the following additional information:

- Whether a CHARACTER, VARCHAR or LONG VARCHAR column was defined as FOR BIT DATA
- Column number
- Partitioning key sequence
- Code page
- Default

For the DESCRIBE INDEXES FOR TABLE command, specifies that output include the following additional information:

- Column names

DESCRIBE

Examples

Describing a SELECT Statement

The following example shows how to describe a SELECT statement:

```
db2 "describe select * from staff"
```

SQLDA Information					
sqldaid :	SQLDA	sqldabc:	896	sqln:	20
sqlid:	7				
Column Information					
sqltype	sqllen	sqlname.data	sqlname.length		
500	SMALLINT	2	ID	2	
449	VARCHAR	9	NAME	4	
501	SMALLINT	2	DEPT	4	
453	CHARACTER	5	JOB	3	
501	SMALLINT	2	YEARS	5	
485	DECIMAL	7, 2	SALARY	6	
485	DECIMAL	7, 2	COMM	4	

Describing a Table

The following example shows how to describe a table:

```
db2 describe table user1.department
```

Table: USER1.DEPARTMENT					
Column name	Type schema	Type name	Length	Scale	Nulls
AREA	SYSIBM	SMALLINT	2	0	No
DEPT	SYSIBM	CHARACTER	3	0	No
DEPTNAME	SYSIBM	CHARACTER	20	0	Yes

DESCRIBE

Describing a Table Index

The following example shows how to describe a table index:

```
db2 describe indexes for table user1.department
```

Table: USER1.DEPARTMENT			
Index schema	Index name	Unique rule	Number of columns
USER1	IDX1	U	2

DETACH

DETACH

Removes the logical DBMS instance attachment, and terminates the physical communication connection if there are no other logical connections using this layer.

Authorization

None

Required Connection

None. Removes an existing instance attachment.

Command Syntax

```
▶▶—DETACH—————▶▶
```

Command Parameters

None

See Also

“ATTACH” on page 91.

DROP DATABASE

Deletes the database contents and all log files for the database, uncatalogs the database, and deletes the database subdirectory.

Scope

By default, this command affects all nodes that are listed in the `db2nodes.cfg` file.

Authorization

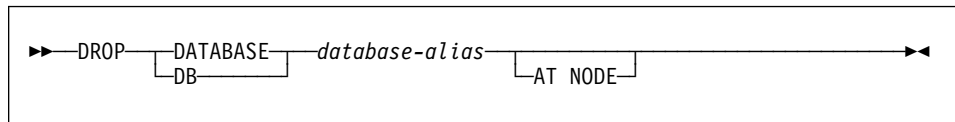
One of the following:

sysadm
sysctrl

Required Connection

Instance. An explicit attachment is not required. If the database is listed as remote, an instance attachment to the remote node is established for the duration of the command.

Command Syntax



Command Parameters

DATABASE *database-alias*

Specifies the alias of the database to be dropped. The database must be cataloged in the system database directory.

AT NODE

Specifies that the database is to be deleted only on the node that issued the DROP DATABASE command. This parameter is used by utilities supplied with DB2 Universal Database Extended Enterprise Edition, and is not intended for general use. Improper use of this parameter can cause inconsistencies in the system, so it should only be used with caution.

Example

The following example deletes the database referenced by the database alias SAMPLE:

```
db2 drop database sample
```

Usage Notes

DROP DATABASE deletes all user data and log files. If the log files are needed for a roll-forward recovery after a restore operation, the files should be saved prior to issuing this command.

DROP DATABASE

The database must not be in use; all users must be disconnected from the database before the database can be dropped.

To be dropped, a database must be cataloged in the system database directory. Only the specified database alias is removed from the system database directory. If other aliases with the same database name exist, their entries remain. If the database being dropped is the last entry in the local database directory, the local database directory is deleted automatically.

If DROP DATABASE is issued from a remote client (or from a different instance on the same machine), the specified alias is removed from the client's system database directory. The corresponding database name is removed from the server's system database directory.

This command unlinks all files that are linked through any DATALINK columns. Since the unlink operation is performed asynchronously on the DB2 File Manager, its effects may not be seen immediately on the DB2 File Manager, and the unlinked files may not be immediately available for other operations. When the command is issued, all the DB2 File Managers configured to that database must be available; otherwise, the drop database operation will fail.

See Also

"CATALOG DATABASE" on page 118

"CREATE DATABASE" on page 143

"UNCATALOG DATABASE" on page 399.

DROP NODE VERIFY

Verifies if a node exists in the nodegroups of any databases, and if an event monitor is defined on the node. This command should be used prior to dropping a node from an MPP system.

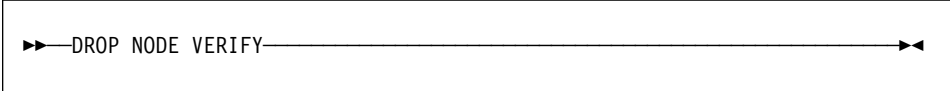
Scope

This command only affects the node on which it is issued.

Authorization

sysadm

Command Syntax



```
▶▶—DROP NODE VERIFY—◀◀
```

Command Parameters

None

Usage Notes

If a message is returned, indicating that the node is not in use, use “STOP DATABASE MANAGER” on page 395 with DROP NODENUM to remove the entry for the node from the `db2nodes.cfg` file, which removes the node from the database system.

If a message is returned, indicating that the node is in use, the following actions should be taken:

1. If the node contains data, redistribute the data to remove it from the node using “REDISTRIBUTE NODEGROUP” on page 337. Use either the DROP NODE option on the REDISTRIBUTE NODEGROUP command, or the ALTER NODEGROUP statement to remove the node from any nodegroups for the database. This must be done for each database that contains the node in a nodegroup. For more information, see the *SQL Reference*.
2. Drop any event monitors that are defined on the node.
3. Rerun DROP NODE VERIFY to ensure that the database is no longer in use.

See Also

“STOP DATABASE MANAGER” on page 395.

ECHO

ECHO

Permits the user to write character strings to standard output.

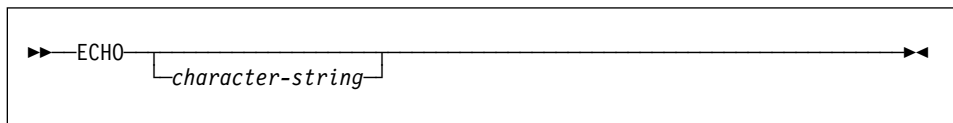
Authorization

None

Required Connection

None

Command Syntax



Command Parameters

character-string

Any character string.

Usage Notes

If an input file is used as standard input, or comments are to be printed without being interpreted by the command shell, the ECHO command will print character strings directly to standard output.

One line is printed each time that ECHO is issued.

The ECHO command is not affected by the verbose (-v) option (see "Command Line Processor Invocation and Options" on page 69).

EXPORT

Exports data from a database to one of several external file formats. The user specifies the data to be exported by supplying an SQL SELECT statement, or by providing hierarchical information for typed tables.

Authorization

One of the following:

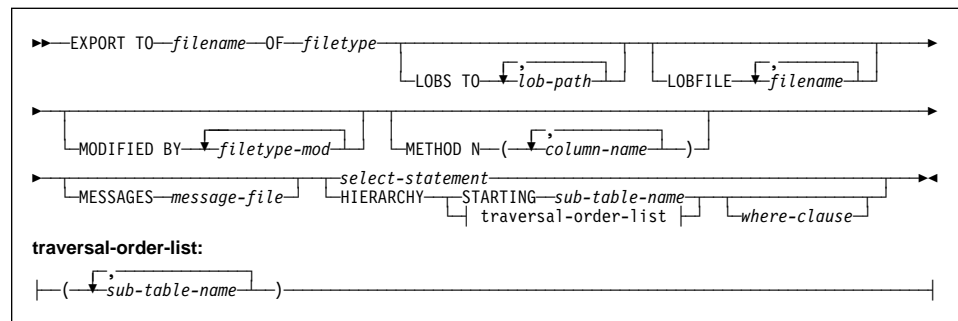
sysadm
dbadm

or CONTROL or SELECT privilege on each participating table or view.

Required Connection

Database. If implicit connect is enabled, a connection to the default database is established.

Command Syntax



Command Parameters

TO *filename*

Specifies the name of the file to which data is to be exported. If the path is omitted, the current working directory is used. If the complete path to the file is not specified, EXPORT uses the current directory and the default drive as the destination.

If the name of a file that already exists is specified, EXPORT overwrites the contents of the file; it does not append the information.

OF *filetype*

Specifies the format of the data in the output file:

- DEL (delimited ASCII format), which is used by a variety of database manager and file manager programs
- WSF (work sheet format), which is used by programs such as:

EXPORT

- Lotus 1-2-3
- Lotus Symphony
- IXF (integrated exchange format, PC version), which makes it possible to import the file again into the same or another DB2 table. Definitions of the table, as well as any existing indexes, are saved in the IXF file, except when columns are specified in the SELECT statement.

For more information about file formats, see the Appendix C, “IMPORT/EXPORT/LOAD Utility File Formats” on page 433.

LOBS TO *lob-path*

Specifies the path or paths to store the LOB files. When file space is exhausted on the first path, the second path will be used, and so on.

LOBFILE *filename*

Specifies the base file name or names of the LOB files. When name space is exhausted for the first name, the second name will be used, and so on.

When creating LOB files during an export, file names are constructed by appending the current base name from this list to the current path (from *lob-path*), and then appending a 3-digit sequence number. For example, if the current LOB path is the directory */u/foo/lob/path*, and the current LOB file name is *bar*, the LOB files created will be */u/foo/lob/path/bar.001*, */u/foo/lob/path/bar.002*, and so on.

MODIFIED BY *filetype-mod*

Specifies additional options (see page 165).

METHOD N *column-name*

Specifies the column name(s) to be used in the output file. If this parameter is not specified, the column names in the existing table are used. This parameter is valid only for WSF and IXF files, but is not valid when exporting hierarchical data.

MESSAGES *message-file*

Specifies the destination for warning and error messages that occur during export. If the file already exists, EXPORT appends the information. If *message-file* is omitted, the messages are written to standard output.

select-statement

Specifies the SELECT statement that will get the information to be exported.

HIERARCHY STARTING *sub-table-name*

Using the default traverse order (OUTER order for ASC, DEL, or WSF files, or the order stored in PC/IXF data files), export a sub-hierarchy starting from *sub-table-name*.

HIERARCHY *traversal-order-list*

Export a sub-hierarchy using the specified traverse order. All sub-tables must be listed in PRE-ORDER fashion. The first sub-table name is used as the target table name for the select statement.

Examples

The following example shows how to export information from the STAFF table in the SAMPLE database (to which the user must be connected) to `myfile.ixf`, with the output in IXF format:

```
db2 export to myfile.ixf of ixf messages msgs.txt select * from staff
```

The following example shows how to export the information about employees in Department 20 from the STAFF table (in the database to which the user must be connected) to `awards.ixf`, with the output in IXF format:

```
db2 export to awards.ixf of ixf messages msgs.txt select * from staff
where dept = 20
```

Usage Notes

Be sure to complete all table operations and release all locks before issuing the EXPORT command. This can be done either by issuing a COMMIT after closing all cursors opened WITH HOLD, or by issuing a ROLLBACK. A COMMIT is performed during the export process.

Table aliases can be used in the SELECT statement.

The messages placed in the message file include the information returned from the message retrieval service. Each message begins on a new line.

The export utility produces a warning message whenever a character column with a length greater than 254 is selected for export to DEL format files.

PC/IXF import should be used to move data between databases. If character data containing row separators is exported to a delimited ASCII (DEL) file and processed by a text transfer program (moving, for example, between OS/2 and AIX systems), fields containing the row separators will shrink or expand.

PC/IXF file format specifications permit migration of data between OS/2 (IBM Extended Services for OS/2, OS/2 Extended Edition and DB2 for OS/2) databases and DB2 for AIX databases via export, binary copying of files between OS/2 and AIX, and import. The file copying step is not necessary if the source and the target databases are both accessible from the same client.

DB2 Connect can be used to export tables from DRDA servers such as DB2 for OS/390, DB2 for VM and VSE, and DB2 for OS/400. Only PC/IXF export is supported.

The EXPORT command will not create multiple-part PC/IXF files when invoked from an AIX system.

When exporting typed tables, subselect statements can only be expressed by specifying the target table name and the WHERE clause. Fullselect and *select-statement* cannot be specified when exporting a hierarchy.

EXPORT

For file formats other than IXF, it is recommended that the traversal order list be specified, because it tells DB2 how to traverse the hierarchy, and what sub-tables to export. If this list is not specified, all tables in the hierarchy are exported, and the default order is the OUTER order. The alternative is to use the default order, which is the order given by the OUTER function.

Note: Use the same traverse order during an import operation. The load utility does not support loading hierarchies or sub-hierarchies.

DB2 File Manager Considerations

To ensure that a consistent copy of the table and the corresponding files referenced by the DATALINK columns are copied for export, do the following:

1. Issue the command: QUIESCE TABLESPACES FOR TABLE tablename SHARE.
This ensures that no update transactions are in progress when EXPORT is run.
2. Issue the EXPORT command.
3. Run the **dlfm_export** utility at each file server (the **dlfm_export** utility is provided with DLFMs). Input to the **dlfm_export** utility is the control file, *server_name*, which is generated by the export utility.
4. Take tar or an equivalent archive of the file list generated by the **dlfm_export** utility.
5. Issue the command: QUIESCE TABLESPACES FOR TABLE tablename RESET.
This makes the table available for updates.

EXPORT is executed as an SQL application. The rows and columns satisfying the SELECT statement conditions are extracted from the database. For the DATALINK columns, the SELECT statement should not specify any scalar function. The export utility uses APIs to extract parts of the DATALINK value, such as link type, file server name, file path name, and comments.

Successful execution of EXPORT results in generation of the following files:

- An export data file as specified in the EXPORT command. A DATALINK column value in this file is in the format described on page 290. When the DATALINK column value is the SQL NULL value, handling is the same as that for other data types.
- Control files *server_name*, which are generated for each file server (on the DOS platform, a single control file, *ctrlfile.lst*, is used by all file servers). A control file contains the URLs for all the files that are to be exported from that file server.

The **dlfm_export** utility is provided to export files from a file server.

```
dlfm_export control_file > filelist
```

Once the *filelist* is obtained, the user should archive the files listed in *filelist*, and restore the archive in the target file server. On UNIX based systems, the following can be done to accomplish this:

1. Make a tape backup from the source file server:

```
tar -cvBf - -L filelist | dd bs=256k of=/dev/rmt0
```

2. Restore from the tape backup to the target file server:

```
dd bs=256k if=/dev/rmt0 | tar xvBf -
```

File Type Modifications

Note: The export utility does not issue a warning if an attempt is made to use unsupported file types with the MODIFIED BY option. If this is attempted, the export fails, and an error code is returned.

Table 5 (Page 1 of 2). Valid File Type Modifications (EXPORT)

Modification	Description
All File Formats	
dldelx	<p><i>x</i> is a single character DATALINK delimiter. The default value is a semicolon (;). The specified character is used in place of a semicolon as the inter-field separator for a DATALINK value. It is needed because a DATALINK value may have more than one sub-value. <i>ab</i></p> <p>Note: For DEL (delimited ASCII) files, <i>x</i> must not be the same character specified as the row, column, or character string delimiter.</p>
lobsinfile	<i>lob-path</i> specifies the path to the files containing LOB values.
DEL (Delimited ASCII) File Format	
chardelx	<p><i>x</i> is a single character string delimiter. The default value is a double quotation mark ("). The specified character is used in place of double quotation marks to enclose a character string.^a</p> <p>The single quotation mark (') can also be specified as a character string delimiter as follows:</p> <p style="text-align: center;">modified by charde1''</p>
coldelx	<p><i>x</i> is a single character column delimiter. The default value is a comma (.). The specified character is used in place of a comma to signal the end of a column.^a</p> <p>In the following example, colde1; causes the export utility to interpret any semicolon (;) it encounters as a column delimiter:</p> <p style="text-align: center;">db2 "export to temp of de1 modified by coldel; select * from staff where dept = 20"</p>
datesiso	Date format. Causes all date data values to be exported in ISO format.
decplusblank	Plus sign character. Causes positive decimal values to be prefixed with a blank space instead of a plus sign (+). The default action is to prefix positive decimal values with a plus sign.

EXPORT

Modification	Description
decptx	x is a single character substitute for the period as a decimal point character. The default value is a period (.). The specified character is used in place of a period as a decimal point character. ^a
nodoubledel	Suppresses recognition of double character delimiters.
WSF File Format	
1	Creates a WSF file that is compatible with Lotus 1-2-3 Release 1, or Lotus 1-2-3 Release 1a. ^b This is the default.
2	Creates a WSF file that is compatible with Lotus Symphony Release 1.0. ^b
3	Creates a WSF file that is compatible with Lotus 1-2-3 Version 2, or Lotus Symphony Release 1.1. ^b
4	Creates a WSF file containing DBCS characters.
<p>Note:</p> <ul style="list-style-type: none"> • ^a Table 6 on page 166 lists the characters that can be used as delimiter overrides. • ^b These files can also be directed to a specific product by specifying an L for Lotus 1-2-3, or an S for Symphony in the <i>filetype-mod</i> parameter string. Only one value or product designator may be specified. 	

Hex	Char	Character Name
X'22'	"	Double Quotation Marks
X'25'	%	Percent Sign
X'26'	&	Ampersand
X'27'	'	Apostrophe
X'28'	(Left Parenthesis
X'29')	Right Parenthesis
X'2A'	*	Asterisk
X'2C'	,	Comma
X'2E'	.	Period (not valid as a character string delimiter)
X'2F'	/	Slash
X'3A'	:	Colon
X'3B'	;	Semicolon
X'3C'	<	Less Than Sign
X'3D'	=	Equals Sign
X'3E'	>	Greater Than Sign

<i>Table 6 (Page 2 of 3). Valid Delimiters</i>		
Hex	Char	Character Name
X'3F'	?	Question Mark
X'7C'		Vertical Bar
X'5F'	_	Underscore (valid in the SBCS environment only)
<p>Note: It is the user's responsibility to ensure that the chosen delimiter character is not part of the data to be moved. If it is, unexpected errors may occur.</p>		
<p>The following restrictions apply to column, string, and decimal point delimiters when moving data:</p> <ul style="list-style-type: none"> • Delimiters are mutually exclusive. • The default decimal point (.) cannot be a string delimiter. • A delimiter cannot be binary zero, a line-feed character, a carriage-return, or a blank space. These characters are specified differently by an ASC-family code page and an EBCDIC-family code page: <ul style="list-style-type: none"> – The Shift-In (0x0F) and the Shift-Out (0x0E) character cannot be delimiters for an EBCDIC MBCS data file. – Delimiters for MBCS, EUC, or DBCS code pages cannot be greater than 0x40, except the default decimal point for EBCDIC MBCS data, which is 0x4b. – Default delimiters for data files in ASC code pages or EBCDIC MBCS code pages are: <ul style="list-style-type: none"> " (0x22, double quotation mark; string delimiter) , (0x2c, comma; column delimiter) – Default delimiters for data files in EBCDIC SBCS code pages are: <ul style="list-style-type: none"> " (0x7F, double quotation mark; string delimiter) , (0x6B, comma; column delimiter) – The default decimal point for ASC data files is 0x2e (period). – The default decimal point for EBCDIC data files is 0x4B (period). <p>If the code page of the server is different from the code page of the client, it is recommended that the hex representation of non-default delimiters be specified. For example,</p> <pre>db2 load from ... modified by charde10x0C colde1X1e ...</pre>		

EXPORT

Table 6 (Page 3 of 3). Valid Delimiters

Hex	Char	Character Name
<p>The following information about support for double character delimiter recognition in DEL files applies to the load, import, and export utilities:</p> <p>Character delimiters are permitted within the character-based fields of a DEL file. Any pair of character delimiters found between the enclosing character delimiters is imported or loaded into the database. For example,</p> <p style="padding-left: 40px;">"What a "nice" day!"</p> <p>will be imported as:</p> <p style="padding-left: 40px;">What a "nice" day!</p> <p>In the case of export, the rule applies in reverse. For example,</p> <p style="padding-left: 40px;">I am 6" tall.</p> <p>will be exported to a DEL file as:</p> <p style="padding-left: 40px;">"I am 6" tall."</p> <p>This support applies to fields of type CHAR, VARCHAR, LONG VARCHAR, or CLOB (except where LOBSINFILE is specified).</p>		

See Also

"IMPORT" on page 219

"LOAD" on page 276.

FORCE APPLICATION

Forces local or remote users or applications off the system to allow for maintenance on a server.

Attention: If an operation that cannot be interrupted (RESTORE DATABASE, for example) is forced, the operation must be successfully re-executed before the database becomes available.

Scope

This command affects all nodes that are listed in the \$HOME/sqllib/db2nodes.cfg file.

In a partitioned database environment, this command does not have to be issued from the coordinator node of the application being forced. It can be issued from any node (database partition server) in the partitioned database environment.

Authorization

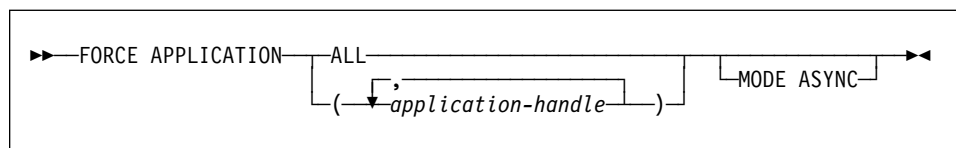
One of the following:

sysadm
sysctrl

Required Connection

Instance. To force users off a remote server, it is first necessary to attach to that server. If no attachment exists, this command is executed locally.

Command Syntax



Command Parameters

APPLICATION

ALL

All applications will be disconnected from the database.

application-handle

Specifies the agent to be terminated. List the values using "LIST APPLICATIONS" on page 236.

MODE ASYNC

The command does not wait for all specified users to be terminated before returning; it returns as soon as the function has been successfully issued or an error (such as invalid syntax) is discovered.

This is the only mode that is currently supported.

FORCE APPLICATION

Example

The following example forces two users, with *application-handle* values of 41408 and 55458, to disconnect from the database:

```
db2 force application ( 41408, 55458 )
```

Usage Notes

db2stop cannot be executed during a force. The database manager remains active so that subsequent database manager operations can be handled without the need for **db2start**.

To preserve database integrity, only users who are idling or executing interruptible database operations can be terminated.

Users creating a database cannot be forced.

After a FORCE has been issued, the database will still accept requests to connect. Additional forces may be required to completely force all users off.

See Also

“ATTACH” on page 91

“LIST APPLICATIONS” on page 236.

GET ADMIN CONFIGURATION

Returns the values of individual entries in the database manager configuration file that are relevant to the DB2 Administration Server. The DB2 Administration Server is a special DB2 instance that enables remote administration of DB2 servers. The following database manager configuration parameters are displayed:

- AGENT_STACK_SZ
- AUTHENTICATION
- DIAGLEVEL
- DIAGPATH
- DISCOVER
- DISCOVER_COMM
- FILESERVER
- IPX_SOCKET
- NNAME
- OBJECTNAME
- QUERY_HEAP_SZ
- SVCENAME
- SYSADM_GROUP
- SYSCTRL_GROUP
- SYSMANT_GROUP
- TPNAME
- TRUST_ALLCLNTS
- TRUST_CLNTAUTH

Note: The SVCENAME parameter, set by the installation program, cannot be modified by the user. The administration server service name is set to use the DB2 registered TCP/IP port (523).

For more information about these parameters, see “GET DATABASE MANAGER CONFIGURATION” on page 188.

Scope

This command returns information on all nodes that share the same \$HOME/sqllib directory, and can be issued from any of these nodes.

Authorization

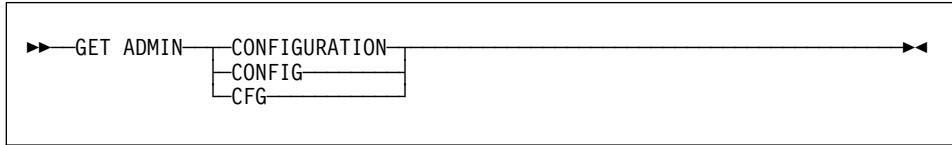
None

Required Connection

None or instance. An instance attachment is not required to perform local DBM configuration operations, but is required to perform remote DBM configuration operations. To display the database manager configuration for a remote instance, it is necessary to first attach to that instance.

GET ADMIN CONFIGURATION

Command Syntax



Command Parameters

None

Example

The following is sample output from GET ADMIN CONFIGURATION:

```
Admin Server Configuration  
  
Node type = Database Server with local clients  
  
Database manager configuration release level          = 0x0800  
  
Diagnostic error capture level          (DIAGLEVEL) = 3  
Diagnostic data directory path          (DIAGPATH) =  
  
SYSADM group name          (SYSADM_GROUP) = BUILD  
SYSCTRL group name          (SYSCTRL_GROUP) =  
SYSMAINT group name          (SYSMAINT_GROUP) =  
  
Database manager authentication          (AUTHENTICATION) = SERVER  
  
Query heap size (4KB)          (QUERY_HEAP_SZ) = 1200  
Discovery mode          (DISCOVER) = KNOWN  
Discovery communication protocols          (DISCOVER_COMM) =
```

Usage Notes

If an attachment to a remote instance (or a different local instance) exists, the admin configuration parameters for the attached server are returned; otherwise, the local admin configuration parameters are returned.

If an error occurs, the information returned is not valid. If the configuration file is invalid, an error message is returned. The user must install the database manager again to recover.

To set the configuration parameters to the default values shipped with the database manager, use “RESET ADMIN CONFIGURATION” on page 352.

GET ADMIN CONFIGURATION

For more information about these parameters, see the *Administration Guide*.

See Also

“RESET ADMIN CONFIGURATION” on page 352

“UPDATE ADMIN CONFIGURATION” on page 405.

GET AUTHORIZATIONS

GET AUTHORIZATIONS

Reports the authorities of the current user from values found in the database configuration file and the authorization system catalog view (SYSCAT.DBAUTH).

Authorization

None

Required Connection

Database. If implicit connect is enabled, a connection to the default database is established.

Command Syntax

```
▶—GET AUTHORIZATIONS—▶
```

Command Parameters

None

Example

The following is sample output from GET AUTHORIZATIONS:

```
Administrative Authorizations for Current User

Direct SYSADM authority           = NO
Direct SYSCTRL authority          = NO
Direct SYSMAINT authority         = NO
Direct DBADM authority            = YES
Direct CREATETAB authority        = YES
Direct BINDADD authority          = YES
Direct CONNECT authority          = YES
Direct CREATE_NOT_FENC authority  = YES
Direct IMPLICIT_SCHEMA authority  = YES

Indirect SYSADM authority         = YES
Indirect SYSCTRL authority        = NO
Indirect SYSMAINT authority       = NO
Indirect DBADM authority          = NO
Indirect CREATETAB authority      = YES
Indirect BINDADD authority        = YES
Indirect CONNECT authority        = YES
Indirect CREATE_NOT_FENC authority = NO
Indirect IMPLICIT_SCHEMA authority = YES
```

GET AUTHORIZATIONS

Usage Notes

Direct authorities are acquired by explicit commands that grant the authorities to a user ID. Indirect authorities are based on authorities acquired by the groups to which a user belongs.

Note: PUBLIC is a special group to which all users belong.

GET CLI CONFIGURATION

GET CLI CONFIGURATION

Lists the contents of the `db2cli.ini` file. This command can list the entire file, or a specified section.

The `db2cli.ini` file is used as the DB2 call level interface (CLI) configuration file. It contains various keywords and values that can be used to modify the behavior of the DB2 CLI and the applications using it. The file is divided into sections, each section corresponding to a database alias name. For more information about this file and the CLI/ODBC configuration keywords, see the *CLI Guide and Reference*.

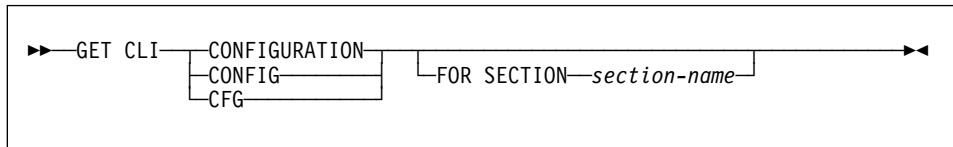
Authorization

None

Required Connection

None

Command Syntax



Command Parameters

FOR SECTION *section-name*

Name of the section whose keywords are to be listed. If not specified, all sections are listed.

Example

The following sample output represents the contents of a `db2cli.ini` file that has two sections:

```
Section: tstcli1x
-----
uid=userid
pwd=*****
autocommit=0
TableType='TABLE','VIEW','SYSTEM TABLE'

Section: tstcli2x
-----
SchemaList='OWNER1','OWNER2',CURRENT SQLID
```

GET CLI CONFIGURATION

Usage Notes

The section name specified on this command is not case sensitive. For example, if the section name in the `db2cli.ini` file (delimited by square brackets) is in lowercase, and the section name specified on the command is in uppercase, the correct section will be listed.

The value of the PWD (password) keyword is never listed; instead, five asterisks (*****) are listed.

See Also

"UPDATE CLI CONFIGURATION" on page 407.

GET CONNECTION STATE

GET CONNECTION STATE

Displays the connection state. Possible states are:

- Connectable and connected
- Connectable and unconnected
- Unconnectable and connected
- Implicitly connectable (if implicit connect is available).

For more information about these connection states, see the *SQL Reference*.

This command also returns information about the database connection mode (SHARE or EXCLUSIVE), and the alias and name of the database to which a connection exists (if one exists).

Authorization

None

Required Connection

None

Command Syntax

```
▶▶ GET CONNECTION STATE ◀◀
```

Command Parameters

None

Example

The following is sample output from GET CONNECTION STATE:

```
Database Connection State
```

```
Connection state      = Connectable and Connected
Connection mode       = SHARE
Local database alias  = SAMPLE
Database name         = SAMPLE
```

Usage Notes

This command does not apply to type 2 connections (see “SET CLIENT” on page 382).

GET DATABASE CONFIGURATION

GET DATABASE CONFIGURATION

Returns the values of individual entries in a specific database configuration file.

Scope

This command returns information only for the node on which it is executed.

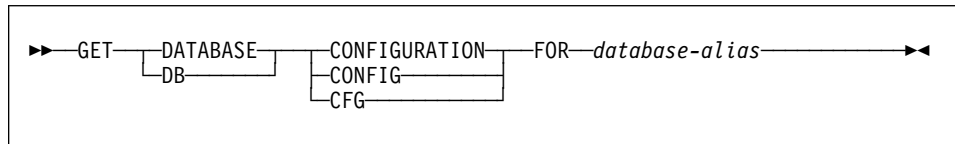
Authorization

None

Required Connection

Instance. An explicit attachment is not required. If the database is listed as remote, an instance attachment to the remote node is established for the duration of the command.

Command Syntax



Command Parameters

FOR *database-alias*

Specifies the alias of the database whose configuration is to be displayed.

Example

Notes:

1. Output on different platforms may show small variations reflecting platform-specific parameters.
2. Parameters with keywords enclosed by parentheses can be changed using "UPDATE DATABASE CONFIGURATION" on page 411.
3. Fields that do not contain keywords are maintained by the database manager and cannot be updated.

The following is sample output from GET DATABASE CONFIGURATION (issued on AIX):

```
Database Configuration for Database sample

Database configuration release level      = 0x0800
Database release level                   = 0x0800

Database territory                        = C
Database code page                        = 819
Database code set                         = ISO8859-1
Database country code                     = 1
```

GET DATABASE CONFIGURATION

```

Directory object name                (DIR_OBJ_NAME) =
Discovery support for this database  (DISCOVER_DB) = ENABLE

Degree of parallelism                (DFT_DEGREE) = 1
Default query optimization class     (DFT_QUERYOPT) = 5
Continue upon arithmetic exceptions  (DFT_SQLMATHWARN) = NO
Number of frequent values retained   (NUM_FREVALUES) = 10
Number of quantiles retained         (NUM_QUANTILES) = 20

Backup pending                       = NO

Database is consistent                = YES
Rollforward pending                  = NO
Restore pending                       = NO

Multi-page file allocation enabled    = NO

Log retain for recovery status        = NO
User exit for logging status          = NO

Datalink Access Token Expiry Interval (sec) (DL_EXPINT) = 1
Datalink Number of Copies             (DL_NUM_COPIES) = 1
Datalink Number of Backups            (DL_NUM_BACKUP) = 1
Datalink Time after Drop (days)      (DL_TIME_DROP) = 1

Database heap (4KB)                  (DBHEAP) = 1200
Catalog cache size (4KB)             (CATALOGCACHE_SZ) = 64
Log buffer size (4KB)                 (LOGBUFSZ) = 8
Utilities heap size (4KB)            (UTIL_HEAP_SZ) = 5000
Buffer pool size (4KB)               (BUFFPAGE) = 1000
Extended storage segments size (4KB) (ESTORE_SEG_SZ) = 16000
Number of extended storage segments   (NUM_ESTORE_SEGS) = 0
Max storage for lock list (4KB)       (LOCKLIST) = 100

Max appl. control heap size (4KB)     (APP_CTL_HEAP_SZ) = 128

Sort list heap (4KB)                 (SORTHEAP) = 256
SQL statement heap (4KB)              (STMTHEAP) = 2048
Default application heap (4KB)        (APPLHEAPSZ) = 128
Package cache size (4KB)             (PCKCACHESZ) = (MAXAPPLS*8)
Statistics heap size (4KB)           (STAT_HEAP_SZ) = 4384

Interval for checking deadlock (ms)   (DLCHKTIME) = 10000
Percent. of lock lists per application (MAXLOCKS) = 10
Lock timeout (sec)                   (LOCKTIMEOUT) = -1

Changed pages threshold               (CHNGPGS_THRESH) = 60
Number of asynchronous page cleaners  (NUM_IOCLEANERS) = 1
Number of I/O servers                 (NUM_IOSERVERS) = 3
Index sort flag                       (INDEXSORT) = YES
Sequential detect flag                (SEQDETECT) = YES
Default prefetch size (4KB)          (DFT_PREFETCH_SZ) = 32

Default number of containers          = 1
Default tablespace extentsize (4KB)  (DFT_EXTENT_SZ) = 32

Max number of active applications     (MAXAPPLS) = 40
Average number of active applications (AVG_APPLS) = 1
Max DB files open per application     (MAXFILOP) = 64

Log file size (4KB)                  (LOGFILSIZ) = 1000
Number of primary log files           (LOGPRIMARY) = 3
Number of secondary log files         (LOGSECOND) = 2
Changed path to log files             (NEWLOGPATH) =
Path to log files                     = /home/user1/user1/NODE0000/SQL00001/SQLLOGDIR/
Next active log file                  =
First active log file                 =

```


GET DATABASE CONFIGURATION

```
Group commit count                (MINCOMMIT) = 1
Percent log file reclaimed before soft ckcpt (SOFTMAX) = 100
Log retain for recovery enabled    (LOGRETAIN) = OFF
User exit for logging enabled      (USEREXIT) = OFF

Auto restart enabled              (AUTORESTART) = ON
Index re-creation time           (INDEXREC) = SYSTEM (RESTART)
Default number of loadrec sessions (DFT_LOADREC_SES) = 1
Recovery history retention (days) (REC_HIS_RETENTN) = 366

ADSM management class            (ADSM_MGMTCLASS) =
ADSM node name                   (ADSM_NODENAME) =
ADSM owner                       (ADSM_OWNER) =
ADSM password                    (ADSM_PASSWORD) =
```

These fields are identified below. Parameters whose name appears in lowercase are maintained by the database manager and cannot be updated. For more information about database configuration parameters, see the *Administration Guide*.

ADSM_MGMTCLASS

The ADSM management class specifies how the server should manage the backup versions or archive copies of the objects being backed up. The management class is assigned from the ADSM administrator. Once assigned, this parameter should be set to the management class name. When performing any ADSM backup, the database manager uses this parameter to pass the management class to ADSM.

ADSM_NODENAME

This parameter is used to override the default setting for the node name associated with the ADSM product. The node name is needed when restoring a database that was backed up to ADSM from another node.

ADSM_OWNER

This parameter is used to override the default setting for the owner associated with the ADSM product. The owner name is needed when restoring a database that was backed up to ADSM from another node.

ADSM_PASSWORD

This parameter is used to override the default setting for the password associated with the ADSM product. The password is needed when restoring a database that was backed up to ADSM from another node.

APP_CTL_HEAP_SZ

This parameter determines the maximum size, in 4KB pages, for the application control heap. The heap is required to share information among agents working on behalf of the same application at a node in an MPP or an SMP system. If complex applications are being run, or the MPP configuration has a large number of nodes, the size of this heap should be increased.

APPLHEAPSZ

Specifies the size, in pages, of the application heap that is available for each individual agent.

AUTORESTART

Indicates whether the database manager can automatically issue RESTART DATABASE on a connect if, for example, the database

GET DATABASE CONFIGURATION

connection was disrupted, or the database was not terminated normally during the previous session.

OFF specifies that it must be done manually.

ON specifies that the database manager does it automatically.

AVG_APPLS

Average number of active applications. Used by the SQL optimizer to help estimate how much buffer pool will be available for the chosen access plan at run time.

backup_pending (Backup pending)

NO specifies that the database is in a usable state.

YES specifies that an offline backup must be performed before the database can be used.

BUFFPAGE

Specifies the size, in pages, of the buffer pool. The buffer pool is used to store and manipulate data read in from the database. This parameter is only used when a buffer pool's size has been explicitly set to -1 through either CREATE BUFFERPOOL, ALTER BUFFERPOOL, or "MIGRATE DATABASE" on page 303. The size of the buffer pool is normally controlled through SQL statements.

CATALOGCACHE_SZ

Controls the size, in pages, of the internal catalog cache (allocated from the *dbheap*), used by the SQL compiler to hold the packed descriptors for commonly referenced objects such as tables and constraints.

CHNGPGS_THRESH

Changed pages threshold. Used to specify the level (percentage) of changed pages at which the asynchronous page cleaners will be started, if they are not currently active.

codepage (Database code page)

Specifies the code page of the database.

codeset (Database code set)

Specifies the code set of the database.

COPYPROTECT (OS/2 only)

Enables the copy-protect attribute.

country (Database country code)

Specifies the country code of the database.

database_consistent (Database is consistent)

NO specifies that a transaction is pending, or some other task is pending on the database, and that the data is not consistent at this point.

YES specifies that all transactions have been committed or rolled back, and that the data is consistent.

database_level (Database release level)

Database release level. Specifies the release level of the database manager which can use the database.

DBHEAP

Specifies the size, in pages, of the database heap that is used to hold control information on all open cursors accessing the database. Both *logbufsz* and *catalogcache_sz* are allocated from the *dbheap*.

GET DATABASE CONFIGURATION

DFT_DEGREE

This parameter specifies the default value for the CURRENT DEGREE special register and the DEGREE bind option.

DFT_EXTENT_SZ

Default extent size of table spaces (in pages).

DFT_LOADREC_SES

Default number of load recovery sessions. Specifies the default number of sessions that will be used during the recovery of a table load. Applicable only if roll-forward recovery is enabled.

DFT_PREFETCH_SZ

Default prefetch size of table spaces (in pages).

DFT_QUERYOPT

The query optimization class is used to direct the optimizer to use different degrees of optimization when compiling SQL queries. This parameter provides additional flexibility by setting the default query optimization class used when neither the SET CURRENT QUERY OPTIMIZATION statement nor the QUERYOPT bind command are used.

DIR_OBJ_NAME

Object name in DCE name space. The object name representing a database manager instance (or a database) in the directory. The concatenation of this value and the *dir_path_name* value yields a global name that uniquely identifies the database manager instance or database in the name space governed by the directory services specified in the *dir_type* parameter.

DISCOVER_DB

This parameter can be set to DISABLE to prevent information about a database from being returned to a client when a discovery request is issued by the client against the server.

DL_EXPINT

Applies to DB2 File Manager only. This parameter specifies the interval of time (in seconds) for which the file access token generated is valid. The number of seconds the token is valid begins from the time it is generated. The File Manager Filter checks the validity of the token containing this expiry time.

DL_NUM_BACKUP

Applies to DB2 File Manager only. This parameter specifies the number of the most recent DB2 backups for which a File Manager keeps backup information. The unlinked files are *garbage collected* based on this value. A value of 1 means that the unlinked files are garbage collected at the completion of the next DB2 backup.

DL_NUM_COPIES

Applies to DB2 File Manager only. This parameter specifies the number of additional copies of a file to be made in the archive server (such as an ADSM server) when a file is linked to the database.

DL_TIME_DROP

Applies to DB2 File Manager only. This parameter specifies the interval of time (in days) files would be retained on an archive server (such as an

GET DATABASE CONFIGURATION

ADSM server) after a DROP TABLE, DROP DATABASE, or DROP TABLESPACE is issued.

DLCHKTIME

Time interval (in milliseconds) for checking deadlock. Defines the frequency at which the database manager checks for deadlocks among all the applications connected to a database.

ESTORE_SEG_SZ

This parameter specifies the number of pages in each of the extended memory segments in the database. There are platform-dependent considerations when setting this configuration parameter.

INDEXREC

Specifies when invalid indexes will be recreated. The default setting is SYSTEM, which uses the value of the database manager configuration parameter *indexrec*.

The possible output values are:

- SYSTEM(ACCESS)
- SYSTEM(RESTART)
- ACCESS
- RESTART.

INDEXSORT

Index sort flag. Indicates whether sorting of index keys will occur during index creation.

LOCKLIST

Specifies the maximum storage, in pages, allocated to the lock list.

LOCKTIMEOUT

Specifies the number of seconds that an application will wait to obtain a lock.

LOGBUFSZ

Specifies the number of pages used to buffer log records prior to writing them to disk. Allocated from *dbheap*.

LOGFILSIZ

Specifies the amount of disk storage, in pages, allocated to log files used for data recovery. This parameter defines the size of each primary and secondary log file.

loghead (First active log file)

Log head identification. Specifies the name of the log file containing the head of the active log. The next log record that is written will start at the head of the active log.

logpath (Path to log files)

Location of log files. Contains the current path being used for logging purposes.

LOGPRIMARY

Specifies the number of primary log files that can be used for database recovery.

LOGRETAIN

Indicates whether the active log files are to be retained and become online archive log files for use in roll-forward recovery (log retention logging).

GET DATABASE CONFIGURATION

log_retain_status (Log retain for recovery status)

Indicates whether log files are being retained for use in roll-forward recovery.

LOGSECOND

Specifies the number of secondary log files that can be used for database recovery.

MAXAPPLS

Specifies the maximum number of application programs (both local and remote) that can connect to the database at one time.

MAXFILOP

Specifies the maximum number of database files that an application program can have open at one time.

MAXLOCKS

Specifies the maximum percentage of the lock list that any one application program can use.

MINCOMMIT

Specifies the number of SQL commits that can be grouped for a given database. Grouping SQL commits permits better control of the I/O and log activity when a commit is performed.

MULTIPAGE_ALLOC

Multi-page file allocation is used to improve insert performance. It applies to SMS table spaces only. If enabled, all SMS table spaces are affected: there is no selection possible for individual SMS table spaces.

NEWLOGPATH

Specifies an alternate path to the recovery log files for a database.

Since the *newlogpath* directory only accepts fully qualified directories, the absolute path must be specified.

nextactive (Next active log file)

Specifies the name of the next recovery log file to be used for logging.

NUM_ESTORE_SEGS

This parameter specifies the number of extended storage memory segments available for use by the database.

NUM_FREQVALUES

Number of frequent values retained. Used to specify the number of "most frequent values" that will be collected when the WITH DISTRIBUTION option is specified in "RUNSTATS" on page 378.

NUM_IOCLEANERS

Specifies the number of asynchronous page cleaners for a database.

NUM_IOSERVERS

Specifies the number of I/O servers for a database. I/O servers are used on behalf of the database agents to perform prefetch I/O and asynchronous I/O by utilities such as backup and restore.

NUM_QUANTILES

Number of quantiles for columns. Controls the number of quantiles that will be collected when the WITH DISTRIBUTION option is specified in "RUNSTATS" on page 378.

GET DATABASE CONFIGURATION

numsegs (Default number of containers)

Determines the number of containers that will be created within the default SMS table spaces.

PCKCACHESZ

Specifies the amount of memory to be used for caching packages and dynamic SQL statements.

The label (calculated) is displayed in the output for “GET DATABASE CONFIGURATION” on page 179 if:

- The internal value is -1
- MAXAPPLS*8 is less than 32. In this case, 32 is displayed with the label (calculated).

REC_HIS_RETENTN

Recovery history retention period. Used to specify the number of days that historical information on backups is to be retained.

RESTORE_PENDING

This parameter indicates whether a RESTORE PENDING status exists in the database.

release (Database configuration release level)

Specifies the release level of the database configuration file.

rollfwd_pending (Rollforward pending)

Indicates whether a roll-forward recovery procedure is required for the database.

The possible values are:

NO

Neither the database nor any table space is in roll-forward pending state.

DATABASE

The database first needs to be rolled forward.

TABLESPACES

One or more table spaces in the database requires roll-forward recovery.

SEQDETECT

Indicates whether sequential detection for a database is to be enabled or disabled.

SOFTMAX

This parameter is used to specify the frequency at which soft checkpoints are taken, and to specify the number of logs that are to be recovered after a crash.

SORTHEAP

Specifies the number of private memory pages available for each sort in the application program.

STAT_HEAP_SZ

Statistics heap size (in pages). Specifies the maximum size of the heap used in creating and collecting all table statistics when distribution statistics are being gathered.

GET DATABASE CONFIGURATION

STMTHEAP

Specifies the heap size, in pages, that can be used for compiling SQL statements.

territory (Database territory)

Specifies the territory of the database.

USEREXIT

Indicates whether a user exit function for archiving or retrieving log files can be called the next time the database is opened.

OFF specifies that a user exit function cannot be called.

ON specifies that a user exit function can be called.

user_exit_status (User exit for logging status)

OFF specifies that the user exit function cannot be called to store archive log files.

ON specifies that the user exit function can be called to store archive log files.

UTIL_HEAP_SZ

Utility heap size. Specifies the maximum amount of shared memory that can be used simultaneously by the backup, restore, and load utilities.

Usage Notes

If an error occurs, the information returned is not valid. If the configuration file is invalid, an error message is returned. The database must be restored from a backup version.

To set the database configuration parameters to the database manager defaults, use "RESET DATABASE CONFIGURATION" on page 354.

For more information about DB2's configuration parameters, see the *Administration Guide*.

See Also

"RESET DATABASE CONFIGURATION" on page 354

"UPDATE DATABASE CONFIGURATION" on page 411.

GET DATABASE MANAGER CONFIGURATION

GET DATABASE MANAGER CONFIGURATION

Returns the values of individual entries in the database manager configuration file.

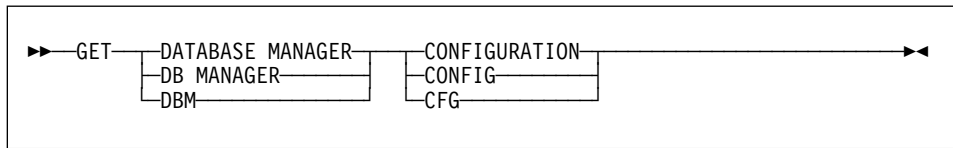
Authorization

None

Required Connection

None or instance. An instance attachment is not required to perform local DBM configuration operations, but is required to perform remote DBM configuration operations. To display the database manager configuration for a remote instance, it is necessary to first attach to that instance.

Command Syntax



Command Parameters

None

Example

Note: Both node type and platform determine which configuration parameters are listed.

The following is sample output from GET DATABASE MANAGER CONFIGURATION (issued on AIX):

```
Database Manager Configuration

Node type = Database Server with local and remote clients

Database manager configuration release level          = 0x0800

CPU speed (millisec/instruction)                    (CPUSPEED) = 4.000000e-05

Max number of concurrently active databases          (NUMDB) = 8
Transaction processor monitor name                   (TP_MON_NAME) =

Default charge-back account                          (DFT_ACCOUNT_STR) =

Java Development Kit 1.1 installation path (JDK11_PATH) =

Diagnostic error capture level                       (DIAGLEVEL) = 3
Diagnostic data directory path                       (DIAGPATH) =

Default database monitor switches
```


GET DATABASE MANAGER CONFIGURATION

```

Buffer pool                                (DFT_MON_BUFPOOL) = OFF
Lock                                        (DFT_MON_LOCK) = OFF
Sort                                        (DFT_MON_SORT) = OFF
Statement                                  (DFT_MON_STMT) = OFF
Table                                       (DFT_MON_TABLE) = OFF
Unit of work                               (DFT_MON_UOW) = OFF

SYSADM group name                          (SYSADM_GROUP) =
SYSCTRL group name                         (SYSCTRL_GROUP) =
SYSMAINT group name                        (SYSMAINT_GROUP) =

Database manager authentication             (AUTHENTICATION) = SERVER
Trust all clients                          (TRUST_ALLCLNTS) = YES
Trusted client authentication              (TRUST_CLNTAUTH) = CLIENT

Default database path                      (DFTDBPATH) = /u/user1

Database monitor heap size (4KB)           (MON_HEAP_SZ) = 48
UDF shared memory set size (4KB)          (UDF_MEM_SZ) = 256
Java Virtual Machine heap size (4KB)      (JAVA_HEAP_SZ) = 512
Audit buffer size (4KB)                   (AUDIT_BUF_SZ) = 0

Backup buffer default size (4KB)          (BACKBUFSZ) = 1024
Restore buffer default size (4KB)         (RESTBUFSZ) = 1024

Sort heap threshold (4KB)                 (SHEAPTHRES) = 20000

Directory cache support                    (DIR_CACHE) = YES

Application support layer heap size (4KB) (ASLHEAPSZ) = 15
Max requester I/O block size (bytes)     (RQRIOBLK) = 32767
Query heap size (4KB)                     (QUERY_HEAP_SZ) = 1000
DRDA services heap size (4KB)             (DRDA_HEAP_SZ) = 128

Priority of agents                          (AGENTPRI) = SYSTEM
Max number of existing agents              (MAXAGENTS) = 200
Agent pool size                            (NUM_POOLAGENTS) = 4 (calculated)
Initial number of agents in pool          (NUM_INITAGENTS) = 0
Max number of coordinating agents         (MAX_COORDAGENTS) = MAXAGENTS
Max no. of concurrent coordinating agents (MAXCAGENTS) = MAX_COORDAGENTS

Keep DARI process                          (KEEPDARI) = YES
Max number of DARI processes               (MAXDARI) = MAX_COORDAGENTS

Index re-creation time                     (INDEXREC) = RESTART

Transaction manager database name          (TM_DATABASE) = TM
Transaction resync interval (sec)         (RESYNC_INTERVAL) = 180

SPM name                                    (SPM_NAME) = user1
SPM log size                               (SPM_LOG_FILE_SZ) = 256
SPM resync agent limit                     (SPM_MAX_RESYNC) = 20
SPM log path                               (SPM_LOG_PATH) = /home/user1/sql1lib

TCP/IP Service name                        (SVCENAME) = xuser1
APPC Transaction program name              (TPNAME) =
IPX/SPX File server name                  (FILESERVER) =
IPX/SPX DB2 server object name            (OBJECTNAME) =
IPX/SPX Socket number                     (IPX_SOCKET) = 879E

```

GET DATABASE MANAGER CONFIGURATION

```
Discovery mode (DISCOVER) = SEARCH
Discovery communication protocols (DISCOVER_COMM) =
Discover server instance (DISCOVER_INST) = ENABLE

Directory services type (DIR_TYPE) = NONE
Directory path name (DIR_PATH_NAME) = /./:/subsys/database/
Directory object name (DIR_OBJ_NAME) =
Routing information object name (ROUTE_OBJ_NAME) =
Default client comm. protocols (DFT_CLIENT_COMM) =

Maximum query degree of parallelism (MAX_QUERYDEGREE) = ANY
Enable intra-partition parallelism (INTRA_PARALLEL) = NO

No. of int. communication buffers(4KB) (FCM_NUM_BUFFERS) = 1024
Number of FCM request blocks (FCM_NUM_RQB) = 512
Number of FCM connection entries (FCM_NUM_CONNECT) = (FCM_NUM_RQB * 0.75)
Number of FCM message anchors (FCM_NUM_ANCHORS) = (FCM_NUM_RQB * 0.75)
```

These fields are identified as follows:

AGENT_STACK_SZ (OS/2 only)

The amount of memory allocated and committed by the operating system for each agent. This parameter specifies the number of pages for each agent stack on the server.

AGENTPRI

Execution priority assigned to database manager processes and threads on a particular machine.

ASLHEAPSZ

Size (in pages) of the memory shared between a local client application and a database manager agent.

AUDIT_BUF_SZ

Size (in pages) of the buffer used when auditing the database.

AUTHENTICATION

Determines how and where authentication of a user takes place. A value of CLIENT indicates that all authentication takes place at the client. If the value is SERVER, the user ID and password are sent from the client to the server so that authentication can take place at the server.

BACKBUFSZ

Size (in pages) of the buffer used when backing up the database, if the buffer size is not specified when calling the backup utility.

CONN_ELAPSE (MPP only)

This parameter specifies the number of seconds within which a TCP/IP connection is to be established between two nodes. If the attempt completes within the time specified by this parameter, communications are established. If it fails, another attempt is made to establish communications. If the connection is attempted the number of times specified by the MAX_CONNRETRIES parameter and always times out, an error is returned.

CPUSPEED

CPU speed (in milliseconds per instruction) used by the SQL optimizer to estimate the cost of performing certain operations. The value of this

GET DATABASE MANAGER CONFIGURATION

parameter is set automatically when the database manager is installed, but can be modified to model a production environment on a test system, or to assess the impact of upgrading hardware.

DFT_ACCOUNT_STR

Default accounting string.

DFT_CLIENT_ADPT (OS/2 only)

This parameter defines the default client adapter number for the NETBIOS protocol whose server nname is extracted from DCE Directory Services.

This parameter can only be used with DCE.

DFT_CLIENT_COMM (DCE only)

Specifies the communication protocols that the client applications on a specific instance can use for remote connections.

DFT_MON_BUFPOOL

Default value of the snapshot monitor's buffer pool switch.

DFT_MON_LOCK

Default value of the snapshot monitor's lock switch.

DFT_MON_SORT

Default value of the snapshot monitor's sort switch.

DFT_MON_STMT

Default value of the snapshot monitor's statement switch.

DFT_MON_TABLE

Default value of the snapshot monitor's table switch.

DFT_MON_UOW

Default value of the snapshot monitor's unit of work (UOW) switch.

DFTDBPATH

Default database path. If no path is specified when a database is created, the database is created on the path specified by this parameter.

DIAGLEVEL

Diagnostic error capture level determines the severity of diagnostic errors recorded in the error log file (db2diag.log).

DIAGPATH

The fully qualified path for DB2 diagnostic information.

DIR_CACHE

Directory cache support. If set to YES, database, node, and DCS directory files are cached in memory. This reduces connect costs by eliminating directory file I/O, and minimizing the directory searches required to retrieve directory information.

DIR_OBJ_NAME

Object name in DCE name space. The object name representing a database manager instance (or a database) in the directory. The concatenation of this value and the *dir_path_name* value yields a global name that uniquely identifies the database manager instance or database in the name space governed by the directory services specified in the *dir_type* parameter.

DIR_PATH_NAME

Directory path name in DCE name space. The unique name of the database manager instance in the global name space is made up of this value and the value in the *dir_obj_name* parameter.

GET DATABASE MANAGER CONFIGURATION

DIR_TYPE

Directory services type. Indicates whether the database manager instance uses the DCE global directory services.

DISCOVER

This parameter defines the type of discovery request supported on a client or server. Discovery requests can be issued from the client configuration assistant or from control center tools. Specify **SEARCH** to support search discovery, in which the DB2 client searches the network for DB2 databases. Specify **KNOWN** to support known discovery, in which the discovery request is issued against the administration server specified by the user. Specify **DISABLE** to disable the client or server from supporting any type of discovery request.

DISCOVER_COMM

This parameter defines the communications protocols that clients use to issue search discovery requests, and servers use to listen for search discovery requests. More than one protocol can be specified, separated by commas, or the parameter can be left blank. Supported protocols are TCP/IP and NETBIOS.

DISCOVER_INST

This parameter enables or disables client discovery of an instance.

DOS_RQRIOLBK

DOS requester I/O block size. Applicable only on DOS clients, including DOS clients running under OS/2. This parameter controls the size of the I/O blocks that are allocated on the client and the server.

DRDA_HEAP_SZ

Specifies the size, in pages, of the DRDA heap. This heap is used by the DRDA AS and by DB2 Connect.

FCM_NUM_ANCHORS

This parameter specifies the number of FCM message anchors. Agents use the message anchors to send messages among themselves.

FCM_NUM_BUFFERS

This parameter specifies the number of 4KB buffers that are used for internal communications (messages) among the nodes in an instance.

FCM_NUM_CONNECT

This parameter specifies the number of FCM connection entries. Agents use connection entries to pass data among themselves.

FCM_NUM_RQB

This parameter specifies the number of FCM request blocks. Request blocks are the media through which information is passed between the FCM daemon and an agent.

FILESERVER

IPX/SPX file server name. Specifies the name of the Novell NetWare file server where the internetwork address of the database manager server instance is registered.

Note: The following characters are not valid: / \ ; , * ?

GET DATABASE MANAGER CONFIGURATION

INDEXREC

Specifies when invalid database indexes should be recreated. This parameter is used if the database configuration parameter *indexrec* is set to SYSTEM.

The possible output values are:

- ACCESS
- RESTART.

INTRA_PARALLEL

This parameter specifies whether the database manager can use intra-partition parallelism.

In a symmetric multiprocessor (SMP) environment, the default for this parameter is YES. In a non-SMP environment, the default for this parameter is NO. This parameter can be used on both partitioned and non-partitioned database systems. Some of the operations that can take advantage of parallel performance improvements when the value of this parameter is YES include database queries and index creation.

IPX_SOCKET

IPX/SPX socket number. Specifies a "well-known" socket number and represents the connection end point in a DB2 server's IPX/SPX internetwork address.

JAVA_HEAP_SZ

Determines the maximum size of the heap that is used by the Java interpreter. For non-partitioned database systems, one heap is allocated for the instance; for partitioned database systems, one heap is allocated for each database partition server.

JDK11_PATH

This parameter specifies the directory under which the Java Development Kit 1.1 is installed. The **CLASSPATH** and other environment variables used by the Java interpreter are computed from the value of this parameter.

KEEPDARI

Indicates whether to keep a DARI process after each DARI call. If NO, a new DARI process is created and terminated for each DARI invocation. If YES, a DARI process is reused for subsequent DARI calls, and is terminated only when the associated user application exits.

MAX_CONNRETRIES (MPP only)

If an attempt to establish communication between two nodes fails because the value specified by the **CONN_ELAPSE** parameter is reached (for example, the attempt to establish TCP/IP communication times out), **MAX_CONNRETRIES** specifies the number of connection retries that can be made to a node. If the value specified for this parameter is exceeded, an error is returned.

MAX_COORDAGENTS

This parameter determines the maximum number of coordinating agents that can exist at one time on a node.

GET DATABASE MANAGER CONFIGURATION

MAX_QUERYDEGREE

This parameter specifies the maximum degree of parallelism used for any SQL statement executing on this instance of the database manager. An SQL statement will not use more than this number of parallel operations when the statement is executed. For a multi-node system, this parameter applies to the degree of parallelism within a single node.

MAX_TIME_DIFF (MPP only)

Each node has its own system clock. This parameter specifies the maximum time difference, in minutes, that is permitted among the nodes listed in the `db2nodes.cfg` file.

MAXAGENTS

Maximum number of database manager agents that can exist simultaneously on a node, regardless of which database is being used.

MAXCAGENTS

Maximum number of database manager agents that can be concurrently executing a database manager transaction. Cannot exceed the value of *maxagents*.

MAXDARI

Maximum number of DARI processes that can reside at the database server. Cannot exceed the value of *maxagents*.

MAXTOTFILOP (OS/2 only)

Maximum number of files open per application. Defines the total database and application file handles that can be used by a specific process connected to a database.

MIN_PRIV_MEM (OS/2 only)

Minimum committed private memory. Specifies the number of pages that the database server process will reserve as private virtual memory when a database manager instance is started (**db2start**).

MON_HEAP_SZ

Database system monitor heap size. Specifies the amount (in 4KB pages) of memory to allocate for database system monitor data.

NNAME (OS/2 only)

Name of the node or workstation. Database clients use *nname* to access database server workstations using NetBIOS. If the database server workstation changes the name specified in *nname*, all clients that access the database server workstation must catalog it again and specify the new name.

nodetype (Node type)

Indicates whether the node is configured as a server with local and remote clients, a client, or a server with local clients.

NUM_INITAGENTS

This parameter determines the initial number of agents that are created in the agent pool when the database manager is started.

NUM_POOLAGENTS

This parameter specifies the size to which the agent pool is allowed to grow. The agent pool contains both idle agents (as in DB2/6000 Version 2), and MPP and SMP associated subagents. If more agents are created, they

GET DATABASE MANAGER CONFIGURATION

will be terminated and not return to the pool when they are finished executing.

If the value of this parameter is calculated at run time using other configuration parameters, the label (calculated) appears to the right of the value shown in the output for "GET DATABASE MANAGER CONFIGURATION" on page 188. If -1 (calculated) is shown in the output, the request was issued from a client, and the value was not available.

The obsolete database manager configuration parameter *max_idleagents* can still be updated through "UPDATE DATABASE MANAGER CONFIGURATION" on page 413, and is interpreted as an update to *num_poolagents*.

NUMDB

Maximum number of local databases that can be concurrently active (that is, have applications connected to them).

OBJECTNAME

This parameter represents the database manager server instance as an object on the NetWare file server, where the server's IPX/SPX address is stored and retrieved. The value must be entered in uppercase. The value must be unique on the NetWare file server, and it is recommended that it be unique across the IPX/SPX network.

Note: The following characters are not valid: / \ ; , * ?

PRIV_MEM_THRESH (OS/2 only)

Private memory threshold. Sets a threshold below which a server will not release the memory associated with a client when that client's connection is terminated.

QUERY_HEAP_SZ

Maximum amount of memory (in pages) that can be allocated for the query heap. A query heap is used to store each query in the agent's private memory.

release (Database manager configuration release level)

Release level of the configuration file.

RESTBUFSZ

Size (in 4KB pages) of the buffer used when restoring the database, if the buffer size is not specified when calling the restore utility.

RESYNC_INTERVAL

Time interval (in seconds) after which a Transaction Manager (TM) or a Resource Manager (RM) retries the recovery of any outstanding indoubt transactions found in the TM or the RM. Applicable when transactions are running in a distributed unit of work (DUOW) environment.

ROUTE_OBJ_NAME (DCE only)

Routing information object name. Specifies the name of the default routing information object entry that will be used by all client applications attempting to access a DRDA server.

GET DATABASE MANAGER CONFIGURATION

RQRIOBLK

Client I/O block size. Specifies the size (in bytes) of the communication buffer between remote applications and their database agents on the database server.

SHEAPTHRESH

Limit on the total amount of memory (in pages) available for sorting across the entire instance.

SPM_NAME

This parameter identifies the name of the Sync Point Manager (SPM) instance to the database manager. The *spm_name* must be defined in the system database directory and, if remote, in the node directory.

SPM_LOG_FILE_SZ

This parameter identifies the Sync Point Manager (SPM) log file size in 4KB pages. The log file is contained in the *spmlog* sub-directory under *sql1ib* and is created the first time SPM is started.

SPM_LOG_PATH

This parameter specifies the directory where the Sync Point Manager (SPM) logs are written. By default, the logs are written to the *sql1ib* directory, which, in a high-volume transaction environment, can cause an I/O bottleneck. Use this parameter to have the SPM log files placed on a faster disk than the current *sql1ib* directory. This allows for better concurrency among the SPM agents.

SPM_MAX_RESYNC

This parameter identifies the number of simultaneous agents that can perform resync operations.

SQLSTMTSZ

Maximum amount of dynamic SQL statement text (in bytes) that will be returned by the database system monitor.

SS_LOGON (OS/2 only)

By accepting the default for this parameter, a LOGON user ID and password are required before issuing a DB2START or DB2STOP.

START_STOP_TIME (MPP only)

This parameter specifies the time, in minutes, within which all nodes must respond to "START DATABASE MANAGER" on page 390, "STOP DATABASE MANAGER" on page 395, or "ADD NODE" on page 89.

SVCENAME

The name used to update the database manager configuration file at the server. This value must be the same as the Connection Service name specified in the *services* file.

SYSADM_GROUP

Defines the group name with system administration (*sysadm*) authority for the database manager instance. This is the highest level of authority within the database manager, and controls all database objects.

SYSCTRL_GROUP

Defines the group name with system control (*sysctrl*) authority for the database manager instance. This level has privileges allowing operations affecting system resources, but not allowing direct access to data.

GET DATABASE MANAGER CONFIGURATION

SYSMANT_GROUP

Defines the group name with system maintenance (*sysmaint*) authority for the database manager instance. This level has authority allowing maintenance operations on all databases associated with an instance, but not allowing direct access to data.

TM_DATABASE

Name of the transaction manager (TM) database for each DB2 instance.

TP_MON_NAME

Name of the transaction processing (TP) monitor product being used.

TPNAME

Name of the remote transaction program that the database client must use when it issues an allocate request to the database manager instance using the APPC communication protocol.

TRUST_ALLCLNTS

This parameter and the TRUST_CLNTAUTH parameter are used to determine where users are validated to the database environment. By accepting the default for this parameter, all clients are treated as trusted clients. This means a level of security is available at the client, and that users can be validated at the client.

TRUST_CLNTAUTH

This parameter and the TRUST_ALLCLNTS parameter are used to determine where users are validated to the database environment. By accepting the default for this parameter, all users of trusted clients are validated at the client.

UDF_MEM_SZ

For a fenced user defined function (UDF), specifies the default allocation for memory to be shared between the database process and the UDF. For an unfenced process, specifies the size of the private memory set. In both cases, this memory is used to pass data to a UDF and back to a database.

Usage Notes

If an attachment to a remote instance (or a different local instance) exists, the database manager configuration parameters for the attached server are returned; otherwise, the local database manager configuration parameters are returned.

If an error occurs, the information returned is invalid. If the configuration file is invalid, an error message is returned. The user must install the database manager again to recover.

To set the configuration parameters to the default values shipped with the database manager, use "RESET DATABASE MANAGER CONFIGURATION" on page 356.

For more information about these parameters, see the *Administration Guide*.

See Also

"RESET DATABASE MANAGER CONFIGURATION" on page 356

"UPDATE DATABASE MANAGER CONFIGURATION" on page 413.

GET DATABASE MANAGER MONITOR SWITCHES

GET DATABASE MANAGER MONITOR SWITCHES

Displays the status of the database system monitor switches. Monitor switches instruct the database system manager to collect database activity information. Each application using the database system monitor interface has its own set of monitor switches (see “GET MONITOR SWITCHES” on page 201). A database manager-level switch is on when any of the monitoring applications has turned it on. This command is used to determine if the database system monitor is currently collecting data for any monitoring application.

Authorization

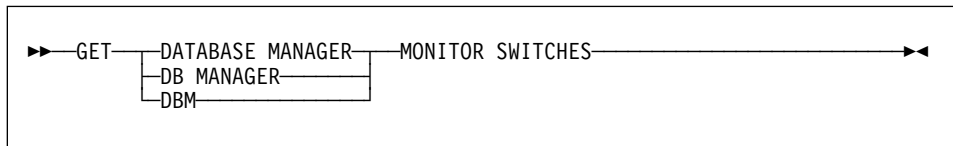
One of the following:

sysadm
sysctrl
sysmaint

Required Connection

Instance. To display the settings for a remote instance, or for a different local instance, it is necessary to first attach to that instance.

Command Syntax



Command Parameters

None

Example

The following is sample output from GET DATABASE MANAGER MONITOR SWITCHES:

```
DBM System Monitor Information Collected
Buffer Pool Activity Information (BUFFERPOOL) = ON 06-11-1997 10:11:01.738377
Lock Information (LOCK) = OFF
Sorting Information (SORT) = ON 06-11-1997 10:11:01.738400
SQL Statement Information (STATEMENT) = OFF
Table Activity Information (TABLE) = OFF
Unit of Work Information (UOW) = ON 06-11-1997 10:11:01.738353
```

GET DATABASE MANAGER MONITOR SWITCHES

Usage Notes

The six recording switches (BUFFERPOOL, LOCK, SORT, STATEMENT, TABLE, and UOW) are off by default, but may be switched on using “UPDATE MONITOR SWITCHES” on page 415. If a particular switch is on, this command also displays the time stamp for when the switch was turned on.

For a summary of all database monitor data elements and monitoring groups, see the *System Monitor Guide and Reference*.

See Also

“GET MONITOR SWITCHES” on page 201

“GET SNAPSHOT” on page 203

“RESET MONITOR” on page 358

“UPDATE MONITOR SWITCHES” on page 415.

GET INSTANCE

GET INSTANCE

Returns the value of the **DB2INSTANCE** environment variable.

Authorization

None

Required Connection

None

Command Syntax

```
▶ GET INSTANCE ▶
```

Command Parameters

None

Example

The following is sample output from GET INSTANCE:

```
The current database manager instance is: smith
```

GET MONITOR SWITCHES

Displays the status of the database system monitor switches for the current session. Monitor switches instruct the database system manager to collect database activity information. Each application using the database system monitor interface has its own set of monitor switches. This command displays them. To display the database manager-level switches, use “GET DATABASE MANAGER MONITOR SWITCHES” on page 198.

Authorization

One of the following:

sysadm
sysctrl
sysmaint

Required Connection

Instance. To display the settings for a remote instance, or for a different local instance, it is necessary to first attach to that instance.

Command Syntax

```
▶ GET MONITOR SWITCHES ◀
```

Command Parameters

None

Example

The following is sample output from GET MONITOR SWITCHES:

```

Monitor Recording Switches

Buffer Pool Activity Information (BUFFERPOOL) = ON 02-20-1997 16:04:30.070073
Lock Information (LOCK) = OFF
Sorting Information (SORT) = OFF
SQL Statement Information (STATEMENT) = ON 02-20-1997 16:04:30.070073
Table Activity Information (TABLE) = OFF
Unit of Work Information (UOW) = ON 02-20-1997 16:04:30.070073

```

Usage Notes

The six recording switches (BUFFERPOOL, LOCK, SORT, STATEMENT, TABLE, and UOW) are off by default, but may be switched on using “UPDATE MONITOR

GET MONITOR SWITCHES

SWITCHES” on page 415. If a particular switch is on, this command also displays the time stamp for when the switch was turned on.

For a summary of all database monitor data elements and monitoring groups, see the *System Monitor Guide and Reference*.

See Also

“GET DATABASE MANAGER MONITOR SWITCHES” on page 198

“GET SNAPSHOT” on page 203

“RESET MONITOR” on page 358

“UPDATE MONITOR SWITCHES” on page 415.

GET SNAPSHOT

Collects database manager status information and returns it to a user-allocated data buffer. The information returned represents a *snapshot* of the database manager operational status at the time the command was issued.

Scope

This command can be invoked from any node in the `db2nodes.cfg` file. It acts only on that node or partition.

Authorization

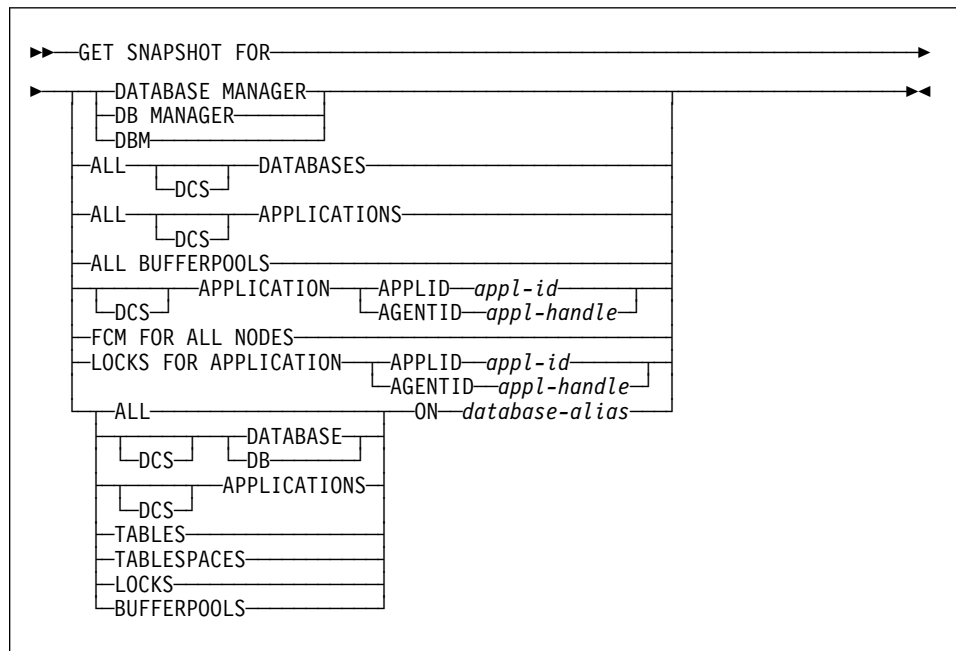
One of the following:

sysadm
sysctrl
sysmaint

Required Connection

Instance. To obtain a snapshot of a remote instance, it is necessary to first attach to that instance.

Command Syntax



Note: The monitor switches must be turned on to get some statistics (see “UPDATE MONITOR SWITCHES” on page 415).

GET SNAPSHOT

Command Parameters

DATABASE MANAGER

Provides statistics for the active database manager instance.

ALL DATABASES

Provides general statistics for all active databases on the current node.

ALL APPLICATIONS

Provides information about all active applications that are connected to a database on the current node.

ALL BUFFERPOOLS

Provides information about buffer pool activity for all active databases.

APPLICATION APPLID *appl-id*

Provides information only about the application whose ID is specified. To get a specific application ID, use "LIST APPLICATIONS" on page 236.

APPLICATION AGENTID *appl-handle*

Provides information only about the application whose application handle is specified. The application handle is a 32-bit number that uniquely identifies an application that is currently running. Use "LIST APPLICATIONS" on page 236 to get a specific application handle.

FCM FOR ALL NODES

Provides FCM statistics for all nodes.

LOCKS FOR APPLICATION APPLID *appl-id*

Provides information about all locks held by the specified application, identified by application ID.

LOCKS FOR APPLICATION AGENTID *appl-handle*

Provides information about all locks held by the specified application, identified by application handle.

ALL ON *database-alias*

Provides general statistics and information about all applications, tables, table spaces, buffer pools, and locks for a specified database.

DATABASE ON *database-alias*

Provides general statistics for a specified database.

APPLICATIONS ON *database-alias*

Provides information about all applications connected to a specified database.

TABLES ON *database-alias*

Provides information about tables in a specified database. This will include only those tables that have been accessed since the TABLE recording switch was turned on.

TABLESPACES ON *database-alias*

Provides information about table spaces for a specified database.

LOCKS ON *database-alias*

Provides information about every lock held by each application connected to a specified database.

BUFFERPOOLS ON *database-alias*

Provides information about buffer pool activity for the specified database.

DCS

Depending on which clause it is specified, this keyword requests statistics about:

GET SNAPSHOT

- A specific DCS application currently running on the DB2 Connect Gateway
- All DCS applications
- All DCS applications currently connected to a specific DCS database
- A specific DCS database
- All DCS databases.

Examples

In the following sample output listings, some of the information may not be available, depending on whether or not the appropriate database system monitor recording switch is turned on (see “UPDATE MONITOR SWITCHES” on page 415). If the information is unavailable, Not Collected appears in the output.

For more information about the fields displayed in the following output listings, see the *System Monitor Guide and Reference*.

The following is typical output resulting from a request for database manager information:

```
Database Manager Snapshot

Node type                = Database Server with local and remote clients
Instance name            = user1
Database manager status  = Active

Product name             =
Product identification    =
Service level            =

Sort heap allocated      = 0
Post threshold sorts     = 0
Piped sorts requested    = 3
Piped sorts accepted     = 3

Start Database Manager timestamp = 04-04-1997 10:54:32.357375
Last reset timestamp     = 04-04-1997 14:28:54.799017
Snapshot timestamp       = 04-06-1997 14:27:19.996209

Remote connections to db manager = 0
Remote connections executing in db manager = 0
Local connections        = 1
Local connections executing in db manager = 0
Active local databases   = 1

High water mark for agents registered = 2
High water mark for agents waiting for a token = 1
Agents registered        = 2
Agents waiting for a token = 0
Idle agents              = 0

Committed private Memory (Bytes) = 12435456

Buffer Pool Activity Information (BUFFERPOOL) = ON 04-04-1997 14:29:42
Lock Information (LOCK) = ON 04-04-1997 14:29:42
Sorting Information (SORT) = ON 04-04-1997 14:29:42
SQL Statement Information (STATEMENT) = ON 04-04-1997 14:29:42
Table Activity Information (TABLE) = ON 04-04-1997 14:29:42
Unit of Work Information (UOW) = ON 04-04-1997 14:29:42
```

GET SNAPSHOT

```
Agents assigned from pool           = 5
Agents created from empty pool      = 2
Agents stolen from another application = 0
High water mark for coordinating agents = 2
Max agents overflow                 = 0
```

The following is typical output resulting from a request for database information:

Database Snapshot

```
Database name                       = SAMPLE
Database path                       = /home/user1/user1/NODE0000/SQL00001/
Input database alias                 =
Database status                     = Active
Catalog node number                 = 0
Catalog network node name           =
Operating system running at database server= AIX
Location of the database             = Local
First database connect timestamp     = 05-15-1998 17:38:50.721558
Last reset timestamp                =
Last backup timestamp               =
Snapshot timestamp                   = 05-15-1998 17:42:50.788090

High water mark for connections     = 1
Application connects                 = 1
Secondary connects total            = 0
Applications connected currently    = 1
Appls. executing in db manager currently = 0
Agents associated with applications = 1
Maximum agents associated with applications= 1
Maximum coordinating agents         = 1

Locks held currently                = 0
Lock waits                          = 0
Time database waited on locks (ms) = 0
Lock list memory in use (Bytes)     = 900
Deadlocks detected                  = 0
Lock escalations                    = 0
Exclusive lock escalations          = 0
Agents currently waiting on locks    = 0
Lock Timeouts                       = 0

Total sort heap allocated           = 0
Total sorts                         = 0
Total sort time (ms)                = 0
Sort overflows                      = 0
Active sorts                        = 0

High water mark for database heap    = 341313

Buffer pool data logical reads      = Not Collected
Buffer pool data physical reads     = Not Collected
Asynchronous pool data page reads  = Not Collected
Buffer pool data writes             = Not Collected
Asynchronous pool data page writes = Not Collected
Buffer pool index logical reads     = Not Collected
Buffer pool index physical reads    = Not Collected
Asynchronous pool index page reads  = Not Collected
Buffer pool index writes            = Not Collected
```

GET SNAPSHOT

```
Asynchronous pool index page writes           = Not Collected
Total buffer pool read time (ms)             = Not Collected
Total buffer pool write time (ms)            = Not Collected
Total elapsed asynchronous read time          = Not Collected
Total elapsed asynchronous write time         = Not Collected
Asynchronous read requests                   = Not Collected
LSN Gap cleaner triggers                     = Not Collected
Dirty page steal cleaner triggers            = Not Collected
Dirty page threshold cleaner triggers        = Not Collected
Time waited for prefetch (ms)               = Not Collected
Direct reads                                 = Not Collected
Direct writes                                = Not Collected
Direct read requests                         = Not Collected
Direct write requests                       = Not Collected
Direct reads elapsed time (ms)              = Not Collected
Direct write elapsed time (ms)              = Not Collected
Database files closed                        = Not Collected
Data pages copied to extended storage         = Not Collected
Index pages copied to extended storage        = Not Collected
Data pages copied from extended storage       = Not Collected
Index pages copied from extended storage      = Not Collected

Commit statements attempted                  = 4
Rollback statements attempted                = 0
Dynamic statements attempted                 = 66
Static statements attempted                  = 4
Failed statement operations                  = 0
Select SQL statements executed                = 3
Update/Insert/Delete statements executed     = 0
DDL statements executed                      = 0

Internal automatic rebinds                   = 0
Internal rows deleted                        = 0
Internal rows inserted                       = 0
Internal rows updated                        = 0
Internal commits                             = 0
Internal rollbacks                           = 0
Internal rollbacks due to deadlock           = 0

Rows deleted                                 = 0
Rows inserted                                = 0
Rows updated                                 = 0
Rows selected                                = 51

Binds/precompiles attempted                 = 0

Maximum secondary log space used (Bytes)     = 0
Maximum total log space used (Bytes)         = 0
Secondary logs allocated currently           = 0
Log pages read                               = 0
Log pages written                            = 0

Package cache lookups                        = 3
Package cache inserts                        = 2
Application section lookups                  = 66
Application section inserts                  = 3

Catalog cache lookups                       = 2
Catalog cache inserts                       = 2
```

GET SNAPSHOT

```
Catalog cache overflows           = 0
Catalog cache heap full           = 0

Number of hash joins               = 0
Number of hash loops               = 0
Number of hash join overflows     = 0
Number of small hash join overflows = 0
```

The following is typical output resulting from a request for DCS database information:

DCS Database Snapshot

```
Database name                      = DCSDB
First database connect timestamp   = 05-21-1998 16:28:39.517919
Last reset timestamp               =
Number of SQL statements attempted = 2
Commit statements attempted        = 1
Rollback statements attempted      = 0
Failed statement operations         = 0
Total number of gateway connections = 0
Current number of gateway connections = 1
Gateway conn. waiting for host reply = 0
Gateway conn. waiting for client reply = 1
High water mark for gateway connections = 1
Rows selected                       = 0
```

The following is typical output resulting from a request for application information (by specifying either an application ID, an application handle, all applications, or all applications on a database):

Application Snapshot

```
Application handle                 = 1
Application status                 = UOW Waiting
Status change time                = 05-15-1998 17:40:52.907292
Application code page              = 819
Application country code           = 1
DUOW correlation token             = *LOCAL.user1.980515213850
Application name                   = db2bp
Application ID                     = *LOCAL.user1.980515213850
Sequence number                   = 0001
Connection request start timestamp = 05-15-1998 17:38:50.721558
Connect request completion timestamp = 05-15-1998 17:38:51.464622
Application idle time              = 1 minute and 10 seconds
Authorization ID                   = USER1
Client login ID                   = user1
Configuration NNAME of client      =
Client database manager product ID = SQL05020
Process ID of client application   = 35544
Platform of client application     = AIX
Communication protocol of client   = Local Client

Outbound communication address     =
Outbound communication protocol   =
Inbound communication address     =

Database name                     = SAMPLE
Database path                      = /home/user1/user1/NODE0000/SQL00001/
```

GET SNAPSHOT

```
Client database alias           = sample
Input database alias           =
Last reset timestamp           =
Snapshot timestamp             = 05-15-1998 17:42:02.285846
The highest authority level granted =
    Indirect SYSADM authority
Coordinating node number       = 0
Current node number            = 0
Coordinator agent process or thread ID = 20668
Agents stolen                   = 0
Agents waiting on locks        = 0
Maximum associated agents       = 1
Priority at which application agents work = 0
Priority type                    = Dynamic

Locks held by application       = 0
Lock waits since connect       = 0
Time application waited on locks (ms) = 0
Deadlocks detected              = 0
Lock escalations               = 0
Exclusive lock escalations     = 0
Number of Lock Timeouts since connected = 0
Total time UOW waited on locks (ms) = 0

Total sorts                     = 0
Total sort time (ms)           = 0
Total sort overflows           = 0

Data pages copied to extended storage = Not Collected
Index pages copied to extended storage = Not Collected
Data pages copied from extended storage = Not Collected
Index pages copied from extended storage = Not Collected
Buffer pool data logical reads    = Not Collected
Buffer pool data physical reads   = Not Collected
Buffer pool data writes           = Not Collected
Buffer pool index logical reads    = Not Collected
Buffer pool index physical reads   = Not Collected
Buffer pool index writes          = Not Collected
Total buffer pool read time (ms)   = Not Collected
Total buffer pool write time (ms)  = Not Collected
Time waited for prefetch (ms)    = Not Collected
Direct reads                      = Not Collected
Direct writes                     = Not Collected
Direct read requests              = Not Collected
Direct write requests             = Not Collected
Direct reads elapsed time (ms)    = Not Collected
Direct write elapsed time (ms)    = Not Collected

Number of SQL requests since last commit = 0
Commit statements                 = 4
Rollback statements               = 0
Dynamic SQL statements attempted  = 66
Static SQL statements attempted   = 4
Failed statement operations       = 0
Select SQL statements executed    = 3
Update/Insert/Delete statements executed = 0
DDL statements executed           = 0
Internal automatic rebinds        = 0
Internal rows deleted             = 0
```

GET SNAPSHOT

```
Internal rows inserted           = 0
Internal rows updated           = 0
Internal commits                 = 0
Internal rollbacks              = 0
Internal rollbacks due to deadlock = 0
Binds/precompiles attempted    = 0
Rows deleted                    = 0
Rows inserted                   = 0
Rows updated                    = 0
Rows selected                   = 51
Rows read                       = 61
Rows written                    = 0

UOW log space used (Bytes)      = 0
Previous UOW completion timestamp = 05-15-1998 17:39:51.885139
UOW start timestamp            = 05-15-1998 17:40:52.859618
UOW stop timestamp             = 05-15-1998 17:40:52.907427
UOW completion status          = Committed - Commit Statement
Open remote cursors            = 0
Open remote cursors with blocking = 0
Rejected Block Remote Cursor requests = 0
Accepted Block Remote Cursor requests = 3
Open local cursors             = 0
Open local cursors with blocking = 0

Total User CPU Time used by agent (s) = 0.160000
Total System CPU Time used by agent (s) = 0.020000
Package cache lookups          = 3
Package cache inserts          = 2
Application section lookups    = 66
Application section inserts    = 3
Catalog cache lookups          = 2
Catalog cache inserts          = 2
Catalog cache overflows        = 0
Catalog cache heap full        = 0

Most recent operation           = Static Commit
Section number                  = 0
Application creator             = NULLID
Package name                    = SQLC28A0
Most recent operation start timestamp = 05-15-1998 17:40:52.906685
Most recent operation stop timestamp = 05-15-1998 17:40:52.907382
Agents associated with the application = 1

Number of hash joins           = 0
Number of hash loops           = 0
Number of hash join overflows  = 0
Number of small hash join overflows = 0

Statement type                  = Static SQL Statement
Statement                       = Static Commit
Section number                  = 0
Application creator             = NULLID
Package name                    = SQLC28A0
Cursor name                     =
Statement node number           = 0
Statement start timestamp       = 05-15-1998 17:40:52.906685
Statement stop timestamp        = 05-15-1998 17:40:52.907382
Total user CPU time             = 0.000000
```

GET SNAPSHOT

```
Total system CPU time           = 0.000000
SQL compiler cost estimate in timerons = 0
SQL compiler cardinality estimate = 0
Degree of parallelism requested = 1
Number of agents working on statement = 0
Number of subagents created for statement = 1
Statement sorts                 = 0
Total sort time                 = 0
Sort overflows                  = 0
Rows read                       = 0
Rows written                    = 0
Rows deleted                    = 0
Rows updated                    = 0
Rows inserted                   = 0
Rows fetched                    = 0
Number of subsections          = 0
```

The following is typical output resulting from a request for DCS application information (by specifying either a DCS application ID, a DCS application handle, all DCS applications, or all DCS applications on a database):

DCS Application Snapshot

```
Client application ID           = 09151251.04D6.980521202839
Sequence number                 = 0001
Authorization ID                = NEWTON
Application name                = db2bp
Application handle              = 0
Application status              = waiting for request
Status change time             = 05-21-1998 16:35:27.670354
Client DB alias                 = MVSDB
Client node                     = antman
Client release level            = SQL05020
Client platform                 = AIX
Client protocol                 = TCP/IP
Client codepage                 = 819
Process ID of client application = 35754
Client login ID                 = user1
Host application ID             = G9151251.G4D7.980521202840
Sequence number                 = 0000
Host DB name                    = GILROY
Host release level              = DSN05011
Host CCSID                      = 500

Outbound communication address   = 9.21.21.92 5021
Outbound communication protocol = TCP/IP
Inbound communication address    = 9.31.12.34 334
First database connect timestamp = 05-21-1998 16:28:39.517919
Time spent on gateway processing = 0.334215
Last reset timestamp            =
Rows selected                   = 0
Number of SQL statements attempted = 2
Failed statement operations      = 0
Commit statements               = 1
Rollback statements             = 0
Inbound bytes received          = 392
Outbound bytes sent             = 136
Outbound bytes received         = 178
```

GET SNAPSHOT

```
Inbound bytes sent                = 190
Number of open cursors            = 0
Application idle time             = 53 seconds

UOW completion status            = Committed - Commit Statement
Previous UOW completion timestamp =
UOW start timestamp              = 05-21-1998 16:35:27.252375
UOW stop timestamp               = 05-21-1998 16:35:27.670290
Inbound bytes received for UOW   = 180
Outbound bytes sent for UOW      = 136
Outbound bytes received for UOW  = 178
Inbound bytes sent for UOW       = 190

Most recent operation            = Static Commit
Most recent operation start timestamp = 05-21-1998 16:35:27.284183
Most recent operation stop timestamp = 05-21-1998 16:35:27.670290

Statement                        = Static Commit
Section number                   = 0
Application creator               = NULLID
Package name                     = SQLC28A0
SQL compiler cost estimate in timerons = 0
SQL compiler cardinality estimate = 0
Statement start timestamp        = 05-21-1998 16:35:27.284183
Statement stop timestamp         = 05-21-1998 16:35:27.670290
Rows fetched                     = 0
Time spent on gateway processing = 0.333740
Inbound bytes received for statement = 0
Outbound bytes sent for statement  = 10
Outbound bytes received for statement = 54
Inbound bytes sent for statement  = 0
```

The following is typical output resulting from a request for buffer pool information:

Bufferpool Snapshot

```
Bufferpool name                  = IBMDEFAULTBP
Database name                    = SAMPLE
Database path                    = /home/user1/user1/NODE0000/SQL00011/
Input database alias             = SAMPLE
Buffer pool data logical reads   = 32
Buffer pool data physical reads  = 13
Buffer pool data writes          = 0
Buffer pool index logical reads  = 55
Buffer pool index physical reads = 23
Total buffer pool read time (ms) = 364
Total buffer pool write time (ms) = 0
Database files closed            = 0

Asynchronous pool data page reads = 0
Asynchronous pool data page writes = 0
Buffer pool index writes         = 0
Asynchronous pool index page reads = 0
Asynchronous pool index page writes = 0
Total elapsed asynchronous read time = 0
Total elapsed asynchronous write time = 0
Asynchronous read requests      = 0
Direct reads                    = 34
Direct writes                    = 0
```


GET SNAPSHOT

```
Direct read requests           = 4
Direct write requests          = 0
Direct reads elapsed time (ms) = 1
Direct write elapsed time (ms) = 0

Data pages copied to extended storage = 0
Index pages copied to extended storage = 0
Data pages copied from extended storage = 0
Index pages copied from extended storage = 0
```

The following is typical output resulting from a request for table information:

Table Snapshot

```
First database connect timestamp = 04-04-1997 14:29:55.197659
Last reset timestamp             =
Snapshot timestamp              = 04-04-1997 14:32:14.151875
Database name                    = SAMPLE
Database path                    = /home/user1/user1/NODE0000/SQL00011/
Input database alias            = SAMPLE
Number of accessed tables       = 6
```

Table Schema	Table Name	Table Type	Rows Written	Rows Read	Overflows
USER1	STAFF	User	0	35	0
USER1	ORG	User	0	8	0
SYSIBM	SYSTABLES	Catalog	0	2	0
SYSIBM	SYSTABLESPACES	Catalog	0	3	0
SYSIBM	SYSPLAN	Catalog	0	1	0
SYSIBM	SYSDBAUTH	Catalog	0	3	0

The following is typical output resulting from a request for table space information:

Tablespace Snapshot

```
First database connect timestamp = 04-04-1997 14:29:55.197659
Last reset timestamp             =
Snapshot timestamp              = 04-04-1997 14:32:14.151875
Database name                    = SAMPLE
Database path                    = /home/user1/user1/NODE0000/SQL00011/
Input database alias            = SAMPLE
Number of accessed tablespaces  = 3
```

```
Tablespace name                 = SYSCATSPACE
```

```
Data pages copied to extended storage = 0
Index pages copied to extended storage = 0
Data pages copied from extended storage = 0
Index pages copied from extended storage = 0
```

```
Buffer pool data logical reads      = 26
Buffer pool data physical reads      = 11
Asynchronous pool data page reads   = 0
Buffer pool data writes              = 0
Asynchronous pool data page writes  = 0
Buffer pool index logical reads      = 55
Buffer pool index physical reads     = 23
Asynchronous pool index page reads   = 0
Buffer pool index writes             = 0
Asynchronous pool index page writes  = 0
Total buffer pool read time (ms)     = 342
Total buffer pool write time (ms)    = 0
Total elapsed asynchronous read time = 0
```

GET SNAPSHOT

```
Total elapsed asynchronous write time = 0
Asynchronous read requests           = 0

Direct reads                          = 34
Direct writes                         = 0
Direct read requests                  = 4
Direct write requests                 = 0
Direct reads elapsed time (ms)       = 1
Direct write elapsed time (ms)       = 0

Number of files closed                = 0

Tablespace name                       = TEMPSPACE1

Data pages copied to extended storage = 0
Index pages copied to extended storage = 0
Data pages copied from extended storage = 0
Index pages copied from extended storage = 0

Buffer pool data logical reads        = 0
Buffer pool data physical reads       = 0
Asynchronous pool data page reads    = 0
Buffer pool data writes               = 0
Asynchronous pool data page writes   = 0
Buffer pool index logical reads       = 0
Buffer pool index physical reads      = 0
Asynchronous pool index page reads   = 0
Buffer pool index writes              = 0
Asynchronous pool index page writes  = 0
Total buffer pool read time (ms)     = 0
Total buffer pool write time (ms)    = 0
Total elapsed asynchronous read time = 0
Total elapsed asynchronous write time = 0
Asynchronous read requests           = 0

Direct reads                          = 0
Direct writes                         = 0
Direct read requests                  = 0
Direct write requests                 = 0
Direct reads elapsed time (ms)       = 0
Direct write elapsed time (ms)       = 0

Number of files closed                = 0

Tablespace name                       = USERSPACE1

Data pages copied to extended storage = 0
Index pages copied to extended storage = 0
Data pages copied from extended storage = 0
Index pages copied from extended storage = 0

Buffer pool data logical reads        = 6
Buffer pool data physical reads       = 2
Asynchronous pool data page reads    = 0
Buffer pool data writes               = 0
Asynchronous pool data page writes   = 0
Buffer pool index logical reads       = 0
Buffer pool index physical reads      = 0
Asynchronous pool index page reads   = 0
```

GET SNAPSHOT

```
Buffer pool index writes           = 0
Asynchronous pool index page writes = 0
Total buffer pool read time (ms)   = 22
Total buffer pool write time (ms)  = 0
Total elapsed asynchronous read time = 0
Total elapsed asynchronous write time = 0
Asynchronous read requests         = 0

Direct reads                       = 0
Direct writes                      = 0
Direct read requests               = 0
Direct write requests              = 0
Direct reads elapsed time (ms)     = 0
Direct write elapsed time (ms)     = 0

Number of files closed              = 0
```

The following is typical output resulting from a request for lock information:

Database Lock Snapshot

```
Database name           = SAMPLE
Database path          = /home/user1/user1/NODE0000/SQL00011/
Input database alias   = SAMPLE
Locks held              = 7
Applications currently connected = 1
Applications currently waiting on locks = 0
Snapshot timestamp     = 04-04-1997 14:32:14.151875

Application handle     = 5
Application ID         = *LOCAL.user1.970404192956
Sequence number       = 0001
Application name       = db2bp_32
Authorization ID       = USER1
Application status     = UOW Waiting
Status change time    =
Application code page = 850
Locks held            = 7
Total wait time (ms)  = 0
```

Object Name	Object Type	Tablespace Name	Table Schema	Table Name	Mode	Status
1545	Row	SYSCATSPACE	SYSIBM	SYSTABLES	NS	Granted
1544	Row	SYSCATSPACE	SYSIBM	SYSTABLES	NS	Granted
2	Table	SYSCATSPACE	SYSIBM	SYSTABLES	IS	Granted
27	Table	SYSCATSPACE	SYSIBM	SYSTABLESPACES	S	Granted
257	Row	SYSCATSPACE	SYSIBM	SYSPLAN	S	Granted
7	Table	SYSCATSPACE	SYSIBM	SYSPLAN	IS	Granted
0	Internal				S	Granted

Usage Notes

To obtain a snapshot from a remote instance (or a different local instance), it is necessary to first attach to that instance. If an alias for a database residing at a different instance is specified, an error message is returned.

To obtain some statistics, it is necessary that the database system monitor switches are turned on.

No data is returned following a request for table information if any of the following is true:

GET SNAPSHOT

- The TABLE recording switch is turned off
- No tables have been accessed since the switch was turned on
- No tables have been accessed since the last RESET MONITOR command was issued.

See Also

“GET MONITOR SWITCHES” on page 201

“LIST APPLICATIONS” on page 236

“RESET MONITOR” on page 358.

HELP

Permits the user to invoke help from the Information Center.

This command is not available on UNIX based systems.

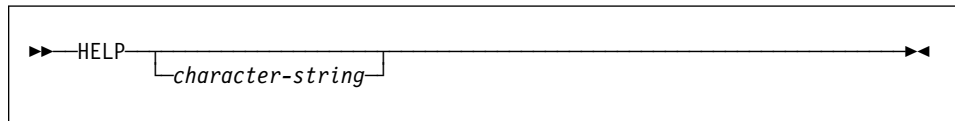
Authorization

None

Required Connection

None

Command Syntax



Command Parameters

HELP *character-string*

Any SQL or DB2 command, or any other item listed in the Information Center.

Examples

Following are examples of the HELP command:

- `db2 help`

This command opens the DB2 Information Center, which contains information about DB2 divided into categories, such as tasks, reference, books, and so on. This is equivalent to invoking the **db2ic** command with no parameters.

- `db2 help drop`

This command opens the Web browser, and displays information about the SQL DROP statement. This is equivalent to invoking the following command: `db2ic -j drop`. The **db2ic** command searches first the *SQL Reference*, and then the *Command Reference*, for a statement or a command called DROP, and then displays the first one found.

- `db2 help 'drop database'`

This command initiates a more refined search, and causes information about the DROP DATABASE command to be displayed.

HELP

Usage Notes

The Information Center must be installed on the user's system. HTML books in the DB2 library must be located in the `\sql11ib\doc\html` subdirectory.

The command line processor will not know if the command succeeds or fails, and cannot report error conditions.

IMPORT

Inserts data from an external file with a supported file format into a table, hierarchy, or view. A faster alternative is “LOAD” on page 276.

Authorization

- IMPORT using the INSERT option requires one of the following:

sysadm

dbadm

CONTROL privilege on each participating table or view

INSERT and SELECT privilege on each participating table or view.

- IMPORT to an existing table using the INSERT_UPDATE, REPLACE, or the REPLACE_CREATE option, requires one of the following:

sysadm

dbadm

CONTROL privilege on the table or view.

- IMPORT to a table or a hierarchy that does not exist using the CREATE, or the REPLACE_CREATE option, requires one of the following:

sysadm

dbadm

CREATETAB authority on the database, and one of:

- IMPLICIT_SCHEMA authority on the database, if the schema name of the table does not exist
- CREATEIN privilege on the schema, if the schema of the table exists.
- CONTROL privilege on every sub-table in the hierarchy, if the REPLACE_CREATE option on the entire hierarchy is used.

- IMPORT to an existing hierarchy using the REPLACE option requires one of the following:

sysadm

dbadm

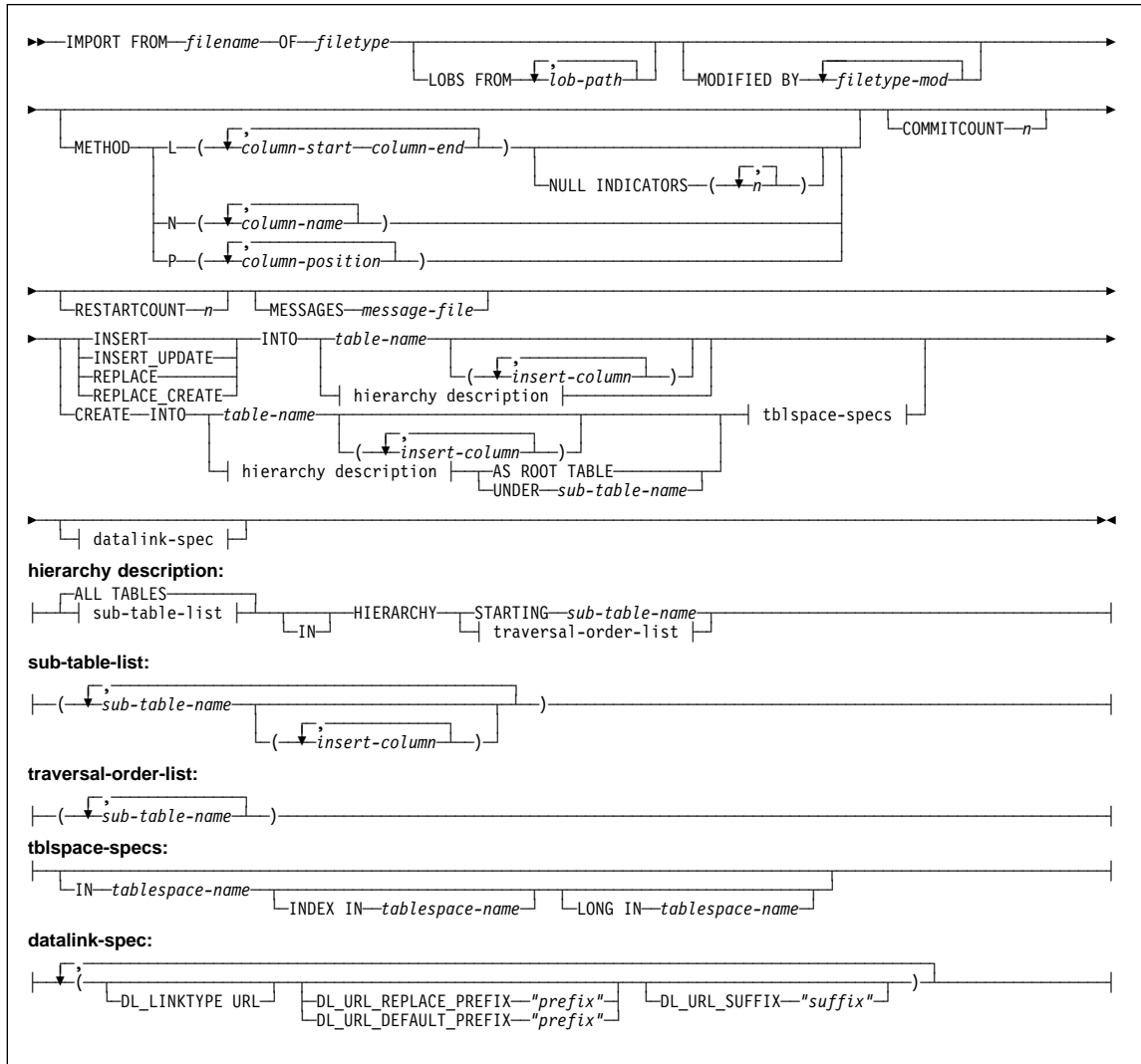
CONTROL privilege on every sub-table in the hierarchy.

Required Connection

Database. If implicit connect is enabled, a connection to the default database is established.

IMPORT

Command Syntax



Command Parameters

FROM *filename*

Specifies the file that contains the data being imported. If the path is omitted, the current working directory is used.

OF *filetype*

Specifies the format of the data in the input file:

- ASC (non-delimited ASCII format)

IMPORT

- DEL (delimited ASCII format), which is used by a variety of database manager and file manager programs
- WSF (work sheet format), which is used by programs such as:
 - Lotus 1-2-3
 - Lotus Symphony
- IXF (integrated exchange format, PC version), which means it was exported from the same or another DB2 table. An IXF file also contains the table definition and definitions of any existing indexes, except when columns are specified in the SELECT statement.

For more information about file formats, see the Appendix C, “IMPORT/EXPORT/LOAD Utility File Formats” on page 433.

LOBS FROM *lob-path*

Specifies the path or paths that store LOB files. The names of the LOB data files are stored in the main data file (ASC, DEL, or IXF), in the column that will be loaded into the LOB column. This option is ignored if *lobsinfile* is not specified within the *filetype-mod* string.

MODIFIED BY *filetype-mod*

Specifies additional information unique to the ASC, DEL, WSF, or IXF file format (see page 228).

METHOD

L

Specifies the start and end column numbers from which to import data.

Note: This method can only be used with ASC files, and is the only valid option for that file type.

N

Specifies the names of the columns to be imported.

Note: This method can only be used with IXF files.

P

Specifies the numbers of the columns to be imported.

Note: This method can only be used with IXF or DEL files, and is the only valid option for the DEL file type.

NULL INDICATORS *n*

Specifies a column (by number) to be used as a null indicator field. If this option is used, a null indicator column for each data column must also be specified. Zero (0) indicates that the data column is not nullable, and that there will always be data in that column.

While processing each row, a Y indicates that the column data is NULL, while an N indicates that the column data is not NULL, and that column data specified by the METHOD L option will be imported.

COMMITCOUNT *n*

Performs a commit every *n* records.

IMPORT

RESTARTCOUNT *n*

Specifies that an import is to be started at record $n + 1$. The first n records are skipped.

MESSAGES *message-file*

Specifies the destination for warning and error messages that occur during import. If the file already exists, IMPORT appends the information. If the complete path to the file is not specified, IMPORT uses the current directory and the default drive as the destination. If *message-file* is omitted, the messages are written to standard output.

INSERT

Adds the imported data to the table without changing the existing table data.

INSERT_UPDATE

Adds rows of imported data to the target table, or updates existing rows (of the target table) with matching primary keys.

REPLACE

Deletes all existing data in the table, and inserts the imported data. The table definition and the index definitions are not changed. This option can only be used if the table exists. It is not valid for tables with DATALINK columns. For typed tables, this option can only operate on the entire hierarchy.

REPLACE_CREATE

If the table exists, deletes all existing data in the table, and inserts the imported data without changing the table definition or the index definitions.

If the table does not exist, creates the table definition and row contents. If the data was exported from a database manager or a DB2 for OS/2 table, indexes are also created.

This option can only be used with IXF files. It is not valid for tables with DATALINK columns. For typed tables, this option can only operate on the entire hierarchy.

CREATE

Creates the table definition and row contents. If the data was exported from a database manager table, sub-table, or hierarchy, indexes are created. If this option operates on a hierarchy, and data was exported from a database manager, a type hierarchy will also be created. This option can only be used with IXF files.

Note: If the data was exported from an MVS host database, and it contains LONGVAR fields whose lengths, calculated on the page size, are less than 254, CREATE may fail because the rows are too long. In this case, the table should be created manually, and IMPORT with INSERT should be invoked, or, alternatively, the LOAD command should be used.

INTO *table-name*

Specifies the database table into which the data is to be imported. This table cannot be a system table or a summary table .

One can use an alias for INSERT, INSERT_UPDATE, or REPLACE, except in the case of a down-level server, when the fully qualified or the unqualified table name should be used. A qualified table name is in the form: *schema.tablename*. The *schema* is the user name under which the table was created.

insert-column

Specifies the name of a column within the table or view into which the data is to be inserted.

ALL TABLES

An implicit keyword for hierarchy only. When importing a hierarchy, the default is to import all tables specified in the traversal order.

sub-table-list

For typed tables with the INSERT or the INSERT_UPDATE option, a list of sub-table names is used to indicate the sub-tables into which data is to be imported.

HIERARCHY

Specifies that hierarchical data is to be imported.

STARTING *sub-table-name*

A keyword for hierarchy only, specifying to use the default order, starting from *sub-table-name*. For PC/IXF files, the default order is the order stored in the data file. The default order is the only valid order for the PC/IXF format.

traversal-order-list

For typed tables with the INSERT, INSERT_UPDATE, or the REPLACE option, a list of sub-table names is used to indicate the traversal order of the importing sub-tables in the hierarchy.

UNDER *sub-table-name*

Specifies a parent table for creating one or more sub-tables.

AS ROOT TABLE

Creates one or more sub-tables as a stand-alone table hierarchy.

IN *tablespace-name*

Identifies the table space in which the table will be created. The table space must exist, and must be a REGULAR table space. If no other table space is specified, all table parts are stored in this table space. If this clause is not specified, the table is created in a table space created by the authorization ID. If none is found, the table is placed into the default table space USERSPACE1. If USERSPACE1 has been dropped, the table creation fails.

INDEX IN *tablespace-name*

Identifies the table space in which any indexes on the table will be created. This option is allowed only when the primary table space specified in the IN clause is a DMS table space. The specified table space must exist, and must be a REGULAR DMS table space.

Note: Specifying which table space will contain a table's index can only be done when the table is created.

IMPORT

LONG IN *tablespace-name*

Identifies the table space in which the values of any long columns (LONG VARCHAR, LONG VARCHARIC, LOB data types, or distinct types with any of these as source types) will be stored. This option is allowed only when the primary table space specified in the IN clause is a DMS table space. The table space must exist, and must be a LONG DMS table space.

DL_LINKTYPE

If specified, it should match the LINKTYPE of the column definition. Thus, DL_LINKTYPE URL is acceptable if the column definition specifies LINKTYPE URL.

DL_URL_DEFAULT_PREFIX *"prefix"*

If specified, it should act as the default prefix for all DATALINK values within the same column. In this context, prefix refers to the "scheme host port" part of the URL specification.

Examples of prefix are:

```
"http://server"  
"file://server"  
"file:"  
"http://server:80"
```

If no prefix is found in a column's data, and a default prefix is specified with DL_URL_DEFAULT_PREFIX, the default prefix is prefixed to the column value (if not NULL).

For example, if DL_URL_DEFAULT_PREFIX specifies the default prefix "http://toronto":

- The column input value "/x/y/z" is stored as "http://toronto/x/y/z".
- The column input value "http://coyote/a/b/c" is stored as "http://coyote/a/b/c".
- The column input value NULL is stored as NULL.

DL_URL_REPLACE_PREFIX *"prefix"*

This clause is useful while loading or importing data previously generated by the export utility, when the user wants to globally replace the host name in the data with another host name. If specified, it becomes the prefix for *all* non-NULL column values. If a column value has a prefix, this will replace it. If a column value has no prefix, the prefix specified by DL_URL_REPLACE_PREFIX is prefixed to the column value.

For example, if DL_URL_REPLACE_PREFIX specifies the prefix "http://toronto":

- The column input value "/x/y/z" is stored as "http://toronto/x/y/z".
- The column input value "http://coyote/a/b/c" is stored as "http://toronto/a/b/c". Note that "toronto" replaces "coyote".
- The column input value NULL is stored as NULL.

DL_URL_SUFFIX *suffix*

If specified, it is appended to every non-NULL column value for the column. It is, in fact, appended to the "path" component of the URL part of the DATALINK value.

Examples

The following example shows how to import information from myfile.ixf to the STAFF table:

```
db2 import from myfile.ixf of ixf messages msg.txt insert into staff
```

```
SQL3150N The H record in the PC/IXF file has product "DB2 01.00", date
"19970220", and time "140848".

SQL3153N The T record in the PC/IXF file has name "ex", qualifier " ",
and source " ".

SQL3109N The utility is beginning to load data from file "ex".

SQL3110N The utility has completed processing. "58" rows were read from the
input file.

SQL3221W ...Begin COMMIT WORK. Input Record Count = "58".

SQL3222W ...COMMIT of any database changes was successful.

SQL3149N "58" rows were processed from the input file. "58" rows were
successfully inserted into the table. "0" rows were rejected.
```

The following example shows how to import the table MOVIE TABLE from the input file delfile1, which has data in the DEL format:

```
db2 import from delfile1 of del
modified by d1del|
insert into movietable (actorname, description, url_making_of, url_movie)
(dl_url_default_prefix "http://narang"),
(dl_url_replace_prefix "http://bomdel" dl_url_suffix ".mpeg")
```

Notes:

1. The table has four columns:

actorname	VARCHAR(n)
description	VARCHAR(m)
url_making_of	DATALINK (with LINKTYPE URL)
url_movie	DATALINK (with LINKTYPE URL)

2. The DATALINK data in the input file has the vertical bar (|) character as the sub-field delimiter.
3. If any column value for url_making_of does not have the prefix character sequence, "http://narang" is used.

IMPORT

4. Each non-NULL column value for `url_movie` will get "http://bomdel" as its prefix. Existing values are replaced.
5. Each non-NULL column value for `url_movie` will get ".mpeg" appended to the path. For example, if a column value of `url_movie` is "http://server1/x/y/z", it will be stored as "http://bomdel/x/y/z.mpeg"; if the value is "/x/y/z", it will be stored as "http://bomdel/x/y/z.mpeg".

Usage Notes

The database table or hierarchy must exist before data in the ASC, DEL, or WSF file formats can be imported; however, if the table does not already exist, `IMPORT CREATE` or `IMPORT REPLACE_CREATE` creates the table when it imports data from a PC/IXF file. For typed tables, `IMPORT CREATE` can create the type hierarchy and the table hierarchy as well. PC/IXF import should be used to move hierarchical data between databases.

The import utility will issue a `COMMIT` or a `ROLLBACK` statement. Ensure that all database activity in the current transaction has completed, and that all locks are released (by issuing a `COMMIT` or a `ROLLBACK`) before issuing the `IMPORT` command.

PC/IXF import should be used to move data between databases. If character data containing row separators is exported to a delimited ASCII (DEL) file and processed by a text transfer program (moving, for example between OS/2 and AIX systems), fields containing the row separators will shrink or expand.

PC/IXF file format specifications permit migration of data between OS/2 (IBM Extended Services for OS/2, OS/2 Extended Edition, and DB2 for OS/2) databases and DB2 for AIX databases via export, binary copying of files between OS/2 and AIX, and import. The file copying step is not necessary if the source and the target databases are both accessible from the same client.

The data in ASC and DEL files is assumed to be in the code page of the client application performing the import.

IXF files, which allow for different code pages, are recommended when importing data in different code pages. If the IXF file and the import utility are in the same code page, processing occurs as for a regular application. If the two differ, and the `FORCEIN` option is specified, `IMPORT` assumes that data in the IXF file has the same code page as the application performing the import. This occurs even if there is a conversion table for the two code pages. If the two differ, `FORCEIN` is not specified, and there is a conversion table, all data in the IXF file will be converted from the file code page to the application code page. If the two differ, `FORCEIN` is not specified, and there is no conversion table, the import will fail. This applies only to IXF files on DB2 for AIX clients.

For table objects on an 8KB page that are close to the limit of 1012 columns, import of IXF data files may cause DB2 to return an error, because the maximum size of an SQL statement was exceeded. This situation can occur only if the columns are of type

CHAR, VARCHAR, or CLOB. The restriction does not apply to import of DEL or ASC files. If IXF files are being used to create a new table, an alternative is to dump the source table's DDL using "db2look - DB2 Statistics Extraction Tool" on page 38, and then to issue that statement through the CLP.

DB2 Connect can be used to import data to DRDA servers such as DB2 for OS/390, DB2 for VM and VSE, and DB2 for OS/400. Only PC/IXF import (INSERT option) is supported. The RESTARTCOUNT parameter, but not the COMMITCOUNT parameter, is also supported.

Importing a multiple-part PC/IXF file whose individual parts are copied from an OS/2 system to an AIX system is supported on DB2.

On the Windows NT operating system:

- Importing logically split IXF files is not supported.
- Importing bad format IXF/WSF files is not supported.

When using the CREATE option with typed tables, create every sub-table defined in the IXF file; sub-table definitions cannot be altered.

When using options other than CREATE with typed tables, the traversal order list enables one to specify the traverse order; therefore, the traversal order list must match the one used during the export operation. For the IXF format, one need only specify the target sub-table name and use the traverse order stored in the file.

DB2 File Manager Considerations

Before running the DB2 import utility, do the following:

1. Copy the DATALINK files into the appropriate file servers.
2. Define one or more prefix names to the DLFMs in the file servers. There may be other administrative tasks, such as registering the database, if required.
3. Update the file server information in the URLs (of the DATALINK columns) from the exported data for the SQL table, if required. (If the original configuration's file servers are the same at the target location, the file server names need not be updated.)
4. Define the file servers at the target configuration in the DB2 File Manager server_configuration file.

When the import utility is executed on the target system, data related to DATALINK columns is loaded into the underlying DB2 tables using SQL INSERT (as is the case for other columns).

The import utility uses APIs to generate values for the DATALINK column. During the insert operation, DATALINK column processing links the file in the appropriate file server according to the column specifications at the target database.

IMPORT

Representation of DATALINK Information in an Input File

For a description of how DATALINK information is represented in an input file, see page 290.

File Type Modifications

Note: The import utility does not issue a warning if an attempt is made to use unsupported file types with the MODIFIED BY option. If this is attempted, the import fails, and an error code is returned.

Table 7 (Page 1 of 4). Valid File Type Modifications (IMPORT)

Modification	Description
All File Formats	
compound= <i>x</i>	<i>x</i> is a number between 1 and 100 inclusive (7 on DOS/Windows). Uses nonatomic compound SQL to insert the data, and <i>x</i> statements will be attempted each time.
dldel <i>x</i>	<i>x</i> is a single character DATALINK delimiter. The default value is a semicolon (;). The specified character is used in place of a semicolon as the inter-field separator for a DATALINK value. It is needed because a DATALINK value may have more than one sub-value. <i>ab</i> Note: For DEL (delimited ASCII) files, <i>x</i> must not be the same character specified as the row, column, or character string delimiter.
lobsinfile	<i>lob-path</i> specifies the path to the files containing LOB values.
no_type_id	Valid only when importing into a single sub-table. Typical usage is to export data from a regular table, and then to invoke an import operation (using this modifier) to convert the data into a single sub-table.
nodefaults	If a source column for a target table column is not explicitly specified, and the table column is not nullable, default values are not loaded. Without this option, if a source column for one of the target table columns is not explicitly specified, one of the following occurs: <ul style="list-style-type: none">• If the column is defaultable, the default value is loaded• If the column is nullable and not defaultable, a NULL is loaded• If the column is not nullable and not defaultable, an error is returned, and the utility stops processing.

<i>Table 7 (Page 2 of 4). Valid File Type Modifications (IMPORT)</i>	
Modification	Description
usedefaults	<p>If a source column for a target table column has been specified, but it contains no data for one or more row instances, default values are loaded. Examples of missing data are:</p> <ul style="list-style-type: none"> • For DEL files: ",," is specified for the column • For DEL/ASC/WSF files: A row that does not have enough columns, or is not long enough for the original specification. <p>Without this option, if a source column contains no data for a row instance, one of the following occurs:</p> <ul style="list-style-type: none"> • If the column is nullable, a NULL is loaded • If the column is not nullable, the utility rejects the row.
ASCII File Formats (ASC/DEL)	
implieddecimal	The location of an implied decimal point is determined by the column definition; it is no longer assumed to be at the end of the value. For example, the value 12345 is loaded into a DECIMAL(8,2) column as 123.45, <i>not</i> 12345.00.
noeofchar	The optional end-of-file character x '1A' is not recognized as the end of file. Processing continues as if it were a normal character.
ASC (Non-delimited ASCII) File Format	
nullindchar=x	<p>x is a single character. Changes the character denoting a null value to x. The default value of x is Y.^b</p> <p>This modifier is case sensitive for EBCDIC data files, except when the character is an English letter. For example, if the null indicator character is specified to be the letter N, then n is also recognized as a null indicator.</p>
reclen=x	x is an integer with a maximum value of 32767. x characters are read for each row, and a new-line character is not used to indicate the end of the row.
striptblanks	<p>Truncates any trailing blank spaces when loading data into a variable-length field. If this option is not specified, blank spaces are kept.</p> <p>In the following example, striptblanks causes the import utility to truncate trailing blank spaces:</p> <pre>db2 import from myfile.asc of asc modified by striptblanks method 1 (1 10, 12 15) messages msgs.txt insert into staff</pre> <p>This option cannot be specified together with striptnulls. These are mutually exclusive options.</p> <p>Note: This option replaces the obsolete t option, which is supported for back-level compatibility only.</p>

IMPORT

<i>Table 7 (Page 3 of 4). Valid File Type Modifications (IMPORT)</i>	
Modification	Description
striptnulls	<p>Truncates any trailing NULLs (0x00 characters) when loading data into a variable-length field. If this option is not specified, NULLs are kept.</p> <p>This option cannot be specified together with striptblanks. These are mutually exclusive options.</p> <p>Note: This option replaces the obsolete padwithzero option, which is supported for back-level compatibility only.</p>
DEL (Delimited ASCII) File Format	
chardelx	<p>x is a single character string delimiter. The default value is a double quotation mark ("). The specified character is used in place of double quotation marks to enclose a character string.^{ab}</p> <p>The single quotation mark (') can also be specified as a character string delimiter. In the following example, charde1'' causes the import utility to interpret any single quotation mark (') it encounters as a character string delimiter:</p> <pre>db2 "import from myfile.del of del modified by charde1'' method p (1, 4) insert into staff (id, years)"</pre>
coldelx	<p>x is a single character column delimiter. The default value is a comma (.). The specified character is used in place of a comma to signal the end of a column.^{ab}</p> <p>In the following example, colde1; causes the import utility to interpret any semicolon (;) it encounters as a column delimiter:</p> <pre>db2 import from myfile.del of del modified by coldel; messages msgs.txt insert into staff</pre>
datesiso	Date format. Causes all date data values to be imported in ISO format.
decplusblank	Plus sign character. Causes positive decimal values to be prefixed with a blank space instead of a plus sign (+). The default action is to prefix positive decimal values with a plus sign.
decptx	<p>x is a single character substitute for the period as a decimal point character. The default value is a period (.). The specified character is used in place of a period as a decimal point character.^{ab}</p> <p>In the following example, decpt; causes the import utility to interpret any semicolon (;) it encounters as a decimal point:</p> <pre>db2 "import from myfile.del of del modified by charde1' decpt; messages msgs.txt insert into staff"</pre>
nodoubledel	Suppresses recognition of double character delimiters.
IXF File Format	

<i>Table 7 (Page 4 of 4). Valid File Type Modifications (IMPORT)</i>	
Modification	Description
defer_import	Valid only for the CREATE option. After the table or sub-tables are created, the import utility returns without importing any data, if this modifier was specified. Typical usage is to invoke the import utility to create the large hierarchy, and then to use the load utility to move the data into the database.
forcein	Directs the utility to accept data despite code page mismatches, and to suppress translation between code pages. Fixed length target fields are checked to verify that they are large enough for the data. If nochecklengths is specified, no checking is done, and an attempt is made to import each row.
indexxf	Directs the utility to drop all indexes currently defined on the existing table, and to create new ones from the index definitions in the PC/IXF file. This option can only be used when the contents of a table are being replaced. It cannot be used with a view, or when a <i>insert-column</i> is specified.
indexschema= <i>schema</i>	Uses the specified <i>schema</i> for the index name during index creation. If <i>schema</i> is not specified (but the keyword <i>indexschema</i> is specified), uses the connection user ID. If the keyword is not specified, uses the schema in the IXF file.
nochecklengths	Used with the forcein modifier. If nochecklengths is specified, fixed length target fields are <i>not</i> checked to verify that they are large enough for the data, and an attempt is made to import each row.
<p>Note:</p> <ul style="list-style-type: none"> ^a Table 6 on page 166 lists the characters that can be used as delimiter overrides. ^b The character must be specified in the code page of the source data. <p>The character code point (instead of the character symbol), can be specified using the syntax xJJ or 0xJJ, where JJ is the hexadecimal representation of the code point. For example, to specify the # character as a column delimiter, use one of the following:</p> <pre> ... modified by coldel# modified by coldel0x23 modified by coldelX23 ... </pre>	

See Also

“EXPORT” on page 161
“LOAD” on page 276.

INITIALIZE TAPE

INITIALIZE TAPE

DB2 for Windows NT supports backup and restore to streaming tape devices. Use this command for tape initialization.

This command is available on Windows NT only.

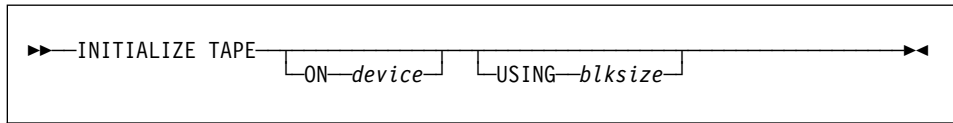
Authorization

None

Required Connection

None

Command Syntax



Command Parameters

ON *device*

Specifies a valid tape device name. The default value is \\.\TAPE0.

USING *blksize*

Specifies the block size for the device. The value must be a multiple of 4KB.

See Also

“REWIND TAPE” on page 369

“SET TAPE POSITION” on page 389.

INVOKE STORED PROCEDURE

INVOKE STORED PROCEDURE

Invokes a procedure stored at the location of a database. Also known as the Database Application Remote Interface (DARI). The server procedure executes at the location of the database, and returns data to the client application.

The application programmer designs the program to run in two parts, one on the client and the other on the server. The server procedure at the database runs within the same transaction as the client application. If the client application and the server procedure are on the same node, the server procedure is executed locally.

Note: This command has been replaced by the SQL CALL statement (see the *SQL Reference*). SQL CALL cannot be used from within the CLP.

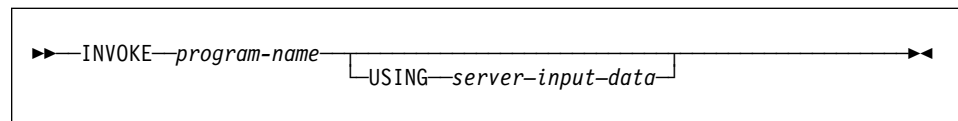
Authorization

CONNECT privilege on a database.

Required Connection

Database

Command Syntax



Note: Do not use INVOKE to call server procedures that use input or output SQLDA structures, including server procedures that return data. For more information, see the *Embedded SQL Programming Guide*.

Command Parameters

program-name

Specifies the procedure to be run on the server. This parameter can be specified in one of the following ways:

- By a *procName* with no extensions. This notation tells the database manager to load a DARI library named *procName* into memory. It is assumed that the name of the function routine to be executed is identical to the library name. For example,

```
db2 invoke foo
```

will load the DARI library named F00 and execute the function routine F00() within the library.

The database manager will find the DARI libraries in the default directory \$HOME/sqllib/function of the instance owner.

- By the exclamation sign (!) delimited name, as in *procName!funcName*. This notation tells the database manager to load

INVOKE STORED PROCEDURE

a DARI library, named `procName`, into memory. The function routine to be executed is `funcName`. This convention allows similar function routines to be packaged in the same DARI library.

- By an absolute path name, as in `/u/cche/procName!/funcName`, for example. This notation includes the storage path of the DARI library. In this example, the DARI library named `procName` is stored in the directory `/u/cche`. The function routine to be executed is `funcName`.

Note: To support portability between various versions of DB2 products, the `!` delimiter can be replaced by the backslash (`\`) delimiter.

USING *server-input-data*

Specifies any information that is passed to the server routine. A variable character string, free form, flexible parameter that can be used to transmit input data according to specific needs.

LIST ACTIVE DATABASES

Displays a subset of the information listed by the GET SNAPSHOT FOR ALL DATABASES command (see “GET SNAPSHOT” on page 203). For each active database, this command displays the following:

- Database name
- Number of applications currently connected to the database
- Database path.

Scope

This command can be issued from any node that is listed in \$HOME/sqllib/db2nodes.cfg. It returns the same information from any of these nodes.

Authorization

None

Command Syntax

```
▶▶—LIST ACTIVE DATABASES—▶▶
```

Command Parameters

None

Examples

Following is sample output from the LIST ACTIVE DATABASES command:

```
Active Databases
Database name           = TEST
Applications connected currently = 0
Database path           = /home/smith/smith/NODE0000/SQL00002/
Database name           = SAMPLE
Applications connected currently = 1
Database path           = /home/smith/smith/NODE0000/SQL00001/
```

See Also

“GET SNAPSHOT” on page 203.

LIST APPLICATIONS

LIST APPLICATIONS

Displays to standard output the application program name, authorization ID (user name), application handle, application ID, and database name of all active database applications. This command can also optionally display an application's sequence number, status, status change time, and database path.

Scope

This command only returns information for the node on which it is issued.

Authorization

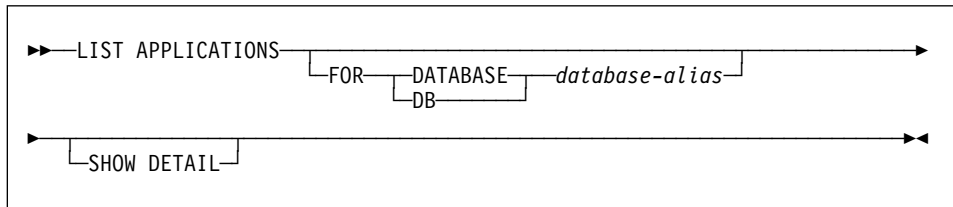
One of the following:

sysadm
sysctrl
sysmaint

Required Connection

Instance. To list applications for a remote instance, it is necessary to first attach to that instance.

Command Syntax



Command Parameters

FOR DATABASE *database-alias*

Information for each application that is connected to the specified database is to be displayed. Database name information is not displayed. If this option is not specified, the command displays the information for each application that is currently connected to any database at the node to which the user is currently attached.

The default application information is comprised of the following:

- Authorization ID
- Application program name
- Application handle
- Application ID
- Database name.

LIST APPLICATIONS

SHOW DETAIL

Output will include the following additional information:

- Sequence #
- Application status
- Status change time
- Database path.

Note: If this option is specified, it is recommended that the output be redirected to a file, and that the report be viewed with the help of an editor. The output lines may wrap around when displayed on the screen.

Example

The following is sample output from LIST APPLICATIONS:

Auth Id	Application Name	Appl. Handle	Application Id	DB Name	# of Agents
smith	db2bp_32	12	*LOCAL.smith.970220191502	TEST	1
smith	db2bp_32	11	*LOCAL.smith.970220191453	SAMPLE	1

Note: For more information about these fields, see the *System Monitor Guide and Reference*.

Usage Notes

The database administrator can use the output from this command as an aid to problem determination. In addition, this information is required if the database administrator wants to use "GET SNAPSHOT" on page 203 or "FORCE APPLICATION" on page 169 in an application.

To list applications at a remote instance (or a different local instance), it is necessary to first attach to that instance. If FOR DATABASE is specified when an attachment exists, and the database resides at an instance which differs from the current attachment, the command will fail.

LIST COMMAND OPTIONS

LIST COMMAND OPTIONS

Lists the current settings for the environment variables:

- **DB2BQTIME**
- **DB2DQTRY**
- **DB2RQTIME**
- **DB2IQTIME**
- **DB2OPTIONS.**

Authorization

None

Required Connection

None

Command Syntax

▶—LIST COMMAND OPTIONS—▶

Command Parameters

None

LIST COMMAND OPTIONS

Example

The following is sample output from LIST COMMAND OPTIONS:

Command Line Processor Option Settings		
Backend process wait time (seconds)		(DB2BQTIME) = 1
No. of retries to connect to backend		(DB2BQTRY) = 60
Request queue wait time (seconds)		(DB2RQTIME) = 5
Input queue wait time (seconds)		(DB2IQTIME) = 5
Command options		(DB2OPTIONS) =
Option	Description	Current Setting
-a	Display SQLCA	OFF
-c	Auto-Commit	ON
-e	Display SQLCODE/SQLSTATE	OFF
-f	Read from input file	OFF
-l	Log commands in history file	OFF
-o	Display output	ON
-p	Display interactive input prompt	ON
-r	Save output to report file	OFF
-s	Stop execution on command error	OFF
-t	Set statement termination character	OFF
-v	Echo current command	OFF
-w	Display FETCH/SELECT warning messages	ON
-z	Save all output to output file	OFF

Usage Notes

For detailed information about these options, see “Command Line Processor Invocation and Options” on page 69.

LIST DATABASE DIRECTORY

LIST DATABASE DIRECTORY

Lists the contents of the system database directory. If a path is specified, the contents of the local database directory are listed.

Scope

If this command is issued without the ON *path* parameter, the system database directory is returned. This information is the same at all nodes.

If the ON *path* parameter is specified, the local database directory on that path is returned. This information is not the same at all nodes.

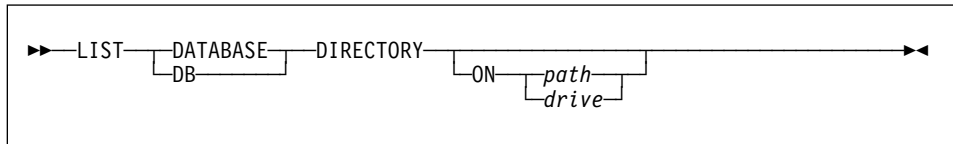
Authorization

None

Required Connection

None. Directory operations affect the local directory only.

Command Syntax



Command Parameters

ON *path/drive*

Specifies the local database directory from which to list information. If not specified, the contents of the system database directory are listed.

LIST DATABASE DIRECTORY

Examples

The following shows sample output for a system database directory:

```
System Database Directory

Number of entries in the directory = 2

Database 1 entry:

Database alias           = SAMPLE
Database name           = SAMPLE
Database drive          = /home/smith
Database release level  = 8.00
Comment                 =
Directory entry type    = Indirect
Catalog node number     = 0

Database 2 entry:

Database alias           = GDB1
Database global name    = ../../cell_name/dir_name/gdb1
Database release level  = 8.00
Comment                 = DCE database
Directory entry type    = DCE
Catalog node number     = -1
```

The following shows sample output for a local database directory:

```
Local Database Directory on /u/smith

Number of entries in the directory = 1

Database 1 entry:

Database alias           = SAMPLE
Database name           = SAMPLE
Database directory      = SQL00001
Database release level  = 8.00
Comment                 =
Directory entry type    = Home
Catalog node number     = 0
Node number             = 0
```

These fields are identified as follows:

Database alias

The value of the *alias* parameter when the database was created or cataloged. If an alias was not entered when the database was cataloged,

LIST DATABASE DIRECTORY

the database manager uses the value of the *database-name* parameter when the database was cataloged.

Database global name

The fully qualified name that uniquely identifies the database in the DCE name space.

Database name

The value of the *database-name* parameter when the database was cataloged. This name is usually the name under which the database was created.

Local database directory

The path on which the database resides. This field is filled in only if the system database directory has been scanned.

Database directory/Database drive

The name of the directory or drive where the database resides. This field is filled in only if the local database directory has been scanned.

Node name

The name of the remote node. This name corresponds to the value entered for the *nodename* parameter when the database and the node were cataloged.

Database release level

The release level of the database manager that can operate on the database.

Comment

Any comments associated with the database that were entered when it was cataloged.

Directory entry type

The location of the database:

- A *remote* entry describes a database that resides on another node.
- An *indirect* entry describes a database that is local. Databases that reside on the same node as the system database directory are thought to indirectly reference the home entry (to a local database directory), and are considered indirect entries.
- A *home* entry indicates that the database directory is on the same path as the local database directory.

All entries in the system database directory are either remote or indirect. All entries in local database directories are identified in the system database directory as indirect entries.

Authentication

The authentication type cataloged at the client is used to determine whether the connection is being done with system or with DCE security.

Catalog node number

Specifies which node is the catalog node (MPP systems only). This is the node on which the CREATE DATABASE command was issued.

LIST DATABASE DIRECTORY

Node number

Specifies the number that is assigned in `db2nodes.cfg` to the node where the command was issued (MPP systems only). In non-MPP systems where there is no `db2nodes.cfg` file, the node number will always be zero.

See Also

"CHANGE DATABASE COMMENT" on page 139

"CREATE DATABASE" on page 143.

LIST DCS APPLICATIONS

LIST DCS APPLICATIONS

Displays to standard output information about applications that are connected to host databases via DB2 Connect Enterprise Edition.

Authorization

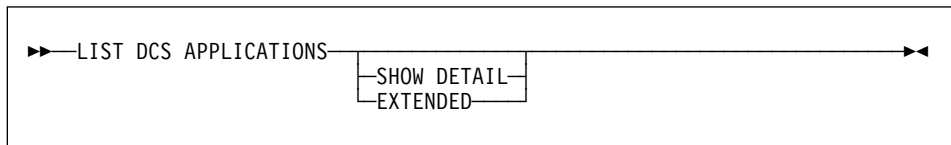
One of the following:

sysadm
sysctrl
sysmaint

Required Connection

Instance. To list the DCS applications at a remote instance, it is necessary to first attach to that instance.

Command Syntax



Command Parameters

LIST DCS APPLICATIONS

The default application information includes:

- Host authorization ID (*username*)
- Application program name
- Application handle
- Outbound application ID (*luwid*).

SHOW DETAIL

Specifies that output include the following additional information:

- Client application ID
- Client sequence number
- Client database alias
- Client node name (*nname*)
- Client release level
- Client code page
- Outbound sequence number
- Host database name
- Host release level.

EXTENDED

Generates an extended report. This report includes all of the fields that are listed when the SHOW DETAIL option is specified, plus the following additional fields:

- DCS application status

LIST DCS APPLICATIONS

- Status change time
- Client platform
- Client protocol
- Client code page
- Process ID of the client application
- Host coded character set ID (CCSID).

Examples

The following is sample output from LIST DCS APPLICATIONS:

Auth Id	Application Name	Appl. Handle	Outbound Application Id
-----	-----	-----	-----
DDCSUS1	db2bp_s	2	0915155C.139D.971205184245

The following is sample output from LIST DCS APPLICATIONS EXTENDED:

List of DCS Applications - Extended Report	
Client application ID	= 09151251.0AD1.980529194106
Sequence number	= 0001
Authorization ID	= SMITH
Application name	= db2bp
Application handle	= 0
Application status	= waiting for reply
Status change time	= Not Collected
Client DB alias	= MVSDB
Client node	= antman
Client release level	= SQL05020
Client platform	= AIX
Client protocol	= TCP/IP
Client codepage	= 819
Process ID of client application	= 38340
Client login ID	= user1
Host application ID	= G9151251.GAD2.980529194108
Sequence number	= 0000
Host DB name	= GILROY
Host release level	= DSN05011
Host CCSID	= 500

Notes:

1. The application status field contains one of the following values:

connect pending - outbound Denotes that the request to connect to a host database has been issued, and that DB2 Connect is waiting for the connection to be established.

LIST DCS APPLICATIONS

waiting for request Denotes that the connection to the host database has been established, and that DB2 Connect is waiting for an SQL statement from the client application.

waiting for reply Denotes that the SQL statement has been sent to the host database.

2. The status change time is shown only if the System Monitor UOW switch was turned on during processing. Otherwise, Not Collected is shown.
3. For more information about these fields, see the *System Monitor Guide and Reference*.

Usage Notes

The database administrator can use this command to match client application connections *to* the gateway with corresponding host connections *from* the gateway.

The database administrator can also use agent ID information to force specified applications off a DB2 Connect server.

See Also

"FORCE APPLICATION" on page 169.

LIST DCS DIRECTORY

Lists the contents of the Database Connection Services (DCS) directory.

Authorization

None

Required Connection

None

Command Syntax

```
▶—LIST DCS DIRECTORY—▶
```

Command Parameters

None

Example

The following is sample output from LIST DCS DIRECTORY:

```

Database Connection Services (DCS) Directory

Number of entries in the directory = 1

DCS 1 entry:

Local database name           = DB2
Target database name         = DSN_DB_1
Application requestor name   =
DCS parameters                =
Comment                       = DB2/MVS Location name DSN_DB_1
DCS directory release level  = 0x0100

```

These fields are identified as follows:

Local database name

Specifies the alias of the target host database. This corresponds to the *database-name* parameter entered when the host database was cataloged in the DCS directory.

LIST DCS DIRECTORY

Target database name

Specifies the name of the host database that can be accessed. This corresponds to the *target-database-name* parameter entered when the host database was cataloged in the DCS directory.

Application requester name

Specifies the name of the program residing on the application requester or server.

DCS parameters

String that contains the connection and operating environment parameters to use with the application requester. Corresponds to the parameter string entered when the host database was cataloged. The string must be enclosed by double quotation marks, and the parameters must be separated by commas.

For more information about DCS parameters, see the *DB2 Connect User's Guide*.

Comment

Describes the database entry.

DCS directory release level

Specifies the version number of the Distributed Database Connection Services program under which the database was created.

Usage Notes

The DCS directory is created the first time that "CATALOG DCS DATABASE" on page 121 is invoked. It is maintained on the path/drive where DB2 was installed, and provides information about host databases that the workstation can access if the DB2 Connect program has been installed. The host databases can be:

- DB2 for MVS/ESA databases on System/370 and System/390 architecture host computers
- DB2 for VSE & VM databases on System/370 and System/390 architecture host computers
- DB2 for AS/400 databases on Application System/400 (AS/400) host computers.

LIST DRDA INDOUBT TRANSACTIONS

Provides a list of transactions that are indoubt between DRDA requesters and DRDA servers. If APPC commit protocols are being used, lists indoubt transactions between partner LUs. If DRDA commit protocols are being used, lists indoubt transactions between DRDA syncpoint managers.

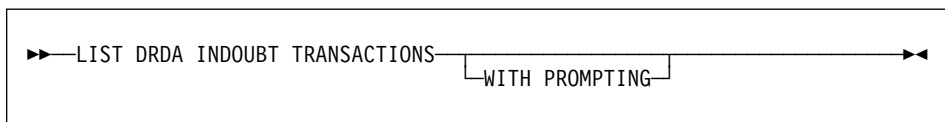
Authorization

sysadm

Required Connection

Instance

Command Syntax



Command Parameters

WITH PROMPTING

Indicates that indoubt transactions are to be processed. If this parameter is specified, an interactive dialog mode is initiated, permitting the user to commit or roll back indoubt transactions. If this parameter is not specified, indoubt transactions are written to the standard output device, and the interactive dialog mode is not initiated.

Note: A forget option is not supported. Once the indoubt transaction is committed or rolled back, the transaction is automatically forgotten.

Interactive dialog mode permits the user to:

- List all indoubt transactions (enter 1)
- List indoubt transaction number *x* (enter 1, followed by a valid transaction number)
- Quit (enter q)
- Commit transaction number *x* (enter c, followed by a valid transaction number)
- Roll back transaction number *x* (enter r, followed by a valid transaction number).

Note: A blank space must separate the command letter from its argument.

Before a transaction is committed or rolled back, the transaction data is displayed, and the user is asked to confirm the action.

LIST DRDA INDOUBT TRANSACTIONS

Usage Notes

DRDA indoubt transactions occur when communication is lost between coordinators and participants in distributed units of work. A distributed unit of work lets a user or application read and update data at multiple locations within a single unit of work. Such work requires a two-phase commit.

The first phase requests all the participants to prepare for a commit. The second phase commits or rolls back the transactions. If a coordinator or participant becomes unavailable after the first phase, the distributed transactions are indoubt.

Before issuing the LIST DRDA INDOUBT TRANSACTIONS command, the application process must be connected to the DB2 Syncpoint Manager (SPM) instance. Use the *spm_name* database manager configuration parameter as the *dbalias* on the CONNECT statement. For more information about using the CONNECT statement, see the *SQL Reference*.

TCP/IP connections, using the SPM to coordinate commits, use DRDA two-phase commit protocols. APPC connections use LU6.2 two-phase commit protocols.

LIST HISTORY

Lists entries in the history file. The history file contains a record of recovery and administrative events. Recovery events include full database and table space level backup, restore, and roll forward. Additional logged events include alter table space, run statistics, reorganize table, drop table, and load.

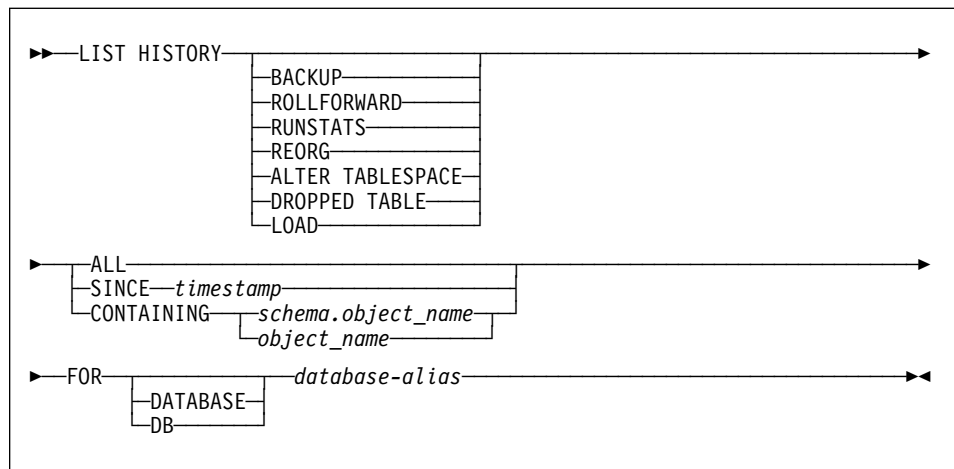
Authorization

None

Required Connection

Instance. An explicit attachment is not required. If the database is listed as remote, an instance attachment to the remote node is established for the duration of the command.

Command Syntax



Command Parameters

HISTORY

Lists all recovery and administration events which are currently logged in the history file.

BACKUP

Lists backup and restore operations.

ROLLFORWARD

Lists roll forward operations.

RUNSTATS

Lists RUNSTATS operations.

REORG

Lists table reorganization operations.

LIST HISTORY

ALTER TABLESPACE

Lists ALTER TABLESPACE operations.

DROPPED TABLE

Lists dropped tables.

LOAD

Lists load operations.

ALL

Lists all entries of the specified type in the history file.

SINCE *timestamp*

A complete time stamp (format *yyyymmddhhnss*), or an initial prefix (minimum *yyyy*) can be specified. All entries with time stamps equal to or less than the time stamp provided are listed.

CONTAINING *schema.object_name*

This qualified name uniquely identifies a table.

CONTAINING *object_name*

This unqualified name uniquely identifies a table space.

FOR DATABASE *database-alias*

Used to identify the database whose recovery history file is to be listed.

Examples

```
db2 list history since 19980201 for sample
db2 list history backup containing userspace1 for sample
db2 list history dropped table all for db sample
```

Usage Notes

The report generated by this command contains the following symbols:

- Device
 - A - ADSM
 - D - Disk
 - K - Diskette
 - O - Other
 - T - Tape
 - U - User Exit
- Object
 - D - Database
 - P - Tablespace
 - T - Table
- Operation
 - B - Backup
 - C - Copy
 - D - Dropped table
 - F - Roll forward
 - G - Reorganize table
 - L - Load
 - R - Restore

LIST HISTORY

- | - S - Run statistics
- | - T - Alter table space

- | • Type

- | - E - End of log
- | - F - Offline
- | - I - Insert(LOAD)
- | - N - Online
- | - P - Point in time
- | - R - Replace (LOAD)

LIST INDOUBT TRANSACTIONS

LIST INDOUBT TRANSACTIONS

Provides a list of transactions that are indoubt. The user can interactively commit, roll back, or forget the indoubt transactions.

The two-phase commit protocol comprises:

1. The PREPARE phase, in which the resource manager writes the log pages to disk, so that it can respond to either a COMMIT or a ROLLBACK primitive
2. The COMMIT (or ROLLBACK) phase, in which the transaction is actually committed or rolled back.

An indoubt transaction is one which has been prepared, but not yet committed or rolled back.

Scope

This command only affects the node on which it is executed.

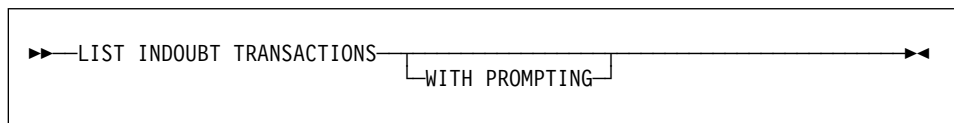
Authorization

dbadm

Required Connection

Database. If implicit connect is enabled, a connection to the default database is established.

Command Syntax



Command Parameters

WITH PROMPTING

Indicates that indoubt transactions are to be processed. If this parameter is specified, an interactive dialog mode is initiated, permitting the user to commit, roll back, or forget indoubt transactions. If this parameter is not specified, indoubt transactions are written to the standard output device, and the interactive dialog mode is not initiated.

Interactive dialog mode permits the user to:

- List all indoubt transactions (enter 1)
- List indoubt transaction number *x* (enter 1, followed by a valid transaction number)
- Quit (enter q)
- Commit transaction number *x* (enter c, followed by a valid transaction number)

LIST INDOUBT TRANSACTIONS

- Roll back transaction number *x* (enter *r*, followed by a valid transaction number)
- Forget transaction number *x* (enter *f*, followed by a valid transaction number).

Note: A blank space must separate the command letter from its argument.

Before a transaction is committed, rolled back, or forgotten, the transaction data is displayed, and the user is asked to confirm the action.

Example

The following is sample dialog generated by LIST INDOUBT TRANSACTIONS:

In-doubt Transactions for Database SAMPLE

```
1.  originator: XA
    appl_id: *LOCAL.DB2.95051815165159      sequence_no: 0001 status: i
timestamp: 05-18-1997 16:51:59 auth_id: SMITH log_full: n type: RM
xid: 53514C2000000017 00000000544D4442 00000000002F93DD A92F8C4FF3000000
0000BD
```

```
2.  originator: XA
    appl_id: *LOCAL.DATABASE.950407161043 sequence_no: 0002 status: i
timestamp: 04-07-1997 16:10:43 auth_id: JONES log_full: n type: RM
xid: 53514C2000000017 00000000544D4442 00000000002F95FE B62F8C4FF3000000
0000C1
```

```
.
.
.
```

Enter in-doubt transaction command or 'q' to quit.

e.g. 'c 1' heuristically commits transaction 1.

```
c/r/f/l/q: c 1
```

```
1.  originator: XA
    appl_id: *LOCAL.DB2.95051815165159      sequence_no: 0001 status: i
timestamp: 05-18-1997 16:51:59 auth_id: SMITH log_full: n type: RM
xid: 53514C2000000017 00000000544D4442 00000000002F93DD A92F8C4FF3000000
0000BD
```

Do you want to heuristically commit this in-doubt transaction ? (y/n) y

```
DB20000I "COMMIT INDOUBT TRANSACTION" completed successfully
```

```
c/r/f/l/q: c 5
```

```
DB20030E "5" is not a valid in-doubt transaction number.
```

```
c/r/f/l/q: l
```

In-doubt Transactions for Database SAMPLE

```
1.  originator: XA
    appl_id: *LOCAL.DB2.95051815165159      sequence_no: 0001 status: c
timestamp: 05-18-1997 16:51:59 auth_id: SMITH log_full: n type: RM
```

LIST INDOUBT TRANSACTIONS

xid: 53514C2000000017 00000000544D4442 00000000002F93DD A92F8C4FF3000000
0000BD

2. originator: XA
app1_id: *LOCAL.DATABASE.950407161043 sequence_no: 0002 status: i
timestamp: 04-07-1997 16:10:43 auth_id: JONES log_full: n type: RM
xid: 53514C2000000017 00000000544D4442 00000000002F95FE B62F8C4FF3000000
0000C1

.

.

c/r/f/l/q: r 2

2. originator: XA
app1_id: *LOCAL.DATABASE.950407161043 sequence_no: 0002 status: i
timestamp: 04-07-1997 16:10:43 auth_id: JONES log_full: n type: RM
xid: 53514C2000000017 00000000544D4442 00000000002F95FE B62F8C4FF3000000
0000C1

Do you want to heuristically rollback this in-doubt transaction ? (y/n) y

DB20000I "ROLLBACK INDOUBT TRANSACTION" completed successfully

c/r/f/l/q: l 2

2. originator: XA
app1_id: *LOCAL.DATABASE.950407161043 sequence_no: 0002 status: r
timestamp: 04-07-1997 16:10:43 auth_id: JONES log_full: n type: RM
xid: 53514C2000000017 00000000544D4442 00000000002F95FE B62F8C4FF3000000
0000C1

c/r/f/l/q: f 2

2. originator: XA
app1_id: *LOCAL.DATABASE.950407161043 sequence_no: 0002 status: r
timestamp: 04-07-1997 16:10:43 auth_id: JONES log_full: n type: RM
xid: 53514C2000000017 00000000544D4442 00000000002F95FE B62F8C4FF3000000
0000C1

Do you want to forget this in-doubt transaction ? (y/n) y

DB20000I "FORGET INDOUBT TRANSACTION" completed successfully

c/r/f/l/q: l 2

2. originator: XA
app1_id: *LOCAL.DATABASE.950407161043 sequence_no: 0002 status: f
timestamp: 04-07-1997 16:10:43 auth_id: JONES log_full: n type: RM
xid: 53514C2000000017 00000000544D4442 00000000002F95FE B62F8C4FF3000000
0000C1

c/r/f/l/q: q

LIST INDOUBT TRANSACTIONS

Usage Notes

An indoubt transaction is a global transaction that was left in an indoubt state. This occurs when either the Transaction Manager (TM) or at least one Resource Manager (RM) becomes unavailable after successfully completing the first phase (that is, the PREPARE phase) of the two-phase commit protocol. The RMs do not know whether to commit or to roll back their branch of the transaction until the TM can consolidate its own log with the indoubt status information from the RMs when they again become available.

If LIST INDOUBT TRANSACTIONS is issued against the currently connected database, the command returns the information on indoubt transactions in that database.

Only transactions whose status is indoubt (i) can be committed.

Only transactions whose status is indoubt (i) or ended (e) can be rolled back.

Only transactions whose status is committed (c) or rolled back (r) can be forgotten.

Indoubt transaction information is valid only at the time that the command is issued. Once in interactive dialog mode, transaction status may change because of external activities. If this happens, and an attempt is made to process an indoubt transaction which is no longer in an appropriate state, an error message is displayed.

After this type of error occurs, the user should quit (q) the interactive dialog and reissue the LIST INDOUBT TRANSACTIONS WITH PROMPTING command to refresh the information shown.

For more information, see the *Administration Guide*.

LIST NODE DIRECTORY

LIST NODE DIRECTORY

Lists the contents of the node directory.

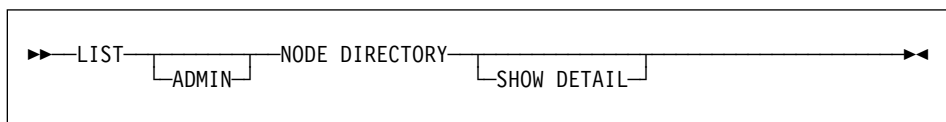
Authorization

None

Required Connection

None

Command Syntax



Command Parameters

ADMIN

Specifies administration server nodes.

SHOW DETAIL

Specifies that the output should include the following information:

- Remote instance name
- System
- Operating system type

Example

The following is sample output from LIST NODE DIRECTORY:

Node Directory

Number of entries in the directory = 6

Node 1 entry:

Node name	= DB2APPC1
Comment	= A remote APPC node
Protocol	= APPC
Symbolic destination name	= db2inst1
Security type	= PROGRAM

Node 2 entry:

Node name	= DB2IPX1
Comment	= A remote IPX/SPX node
Protocol	= IPXSPX
File server name	= netwsrv
Bindery object name	= db2inst1

LIST NODE DIRECTORY

Node 3 entry:

Node name	= DB2IPX2
Comment	= IPX/SPX node using direct addr
Protocol	= IPXSPX
File server name	= *
Bindery object name	= 09212700.400011527745.879E

Node 4 entry:

Node name	= DB2NETB1
Comment	= A remote NetBIOS node
Protocol	= NETBIOS
Adapter number	= 0
Server NNAME	= DB2INST1

Node 5 entry:

Node name	= DB2TCP1
Comment	= A remote TCP/IP node
Protocol	= TCPIP
Hostname	= tcphost
Service name	= db2inst1

Node 6 entry:

Node name	= DB2TCP2
Comment	= TCP/IP node using IP address
Protocol	= TCPIP
Hostname	= 9.21.15.235
Service name	= db2inst2

The common fields are identified as follows:

Node name

The name of the remote node. This corresponds to the name entered for the *nodename* parameter when the node was cataloged.

Comment

A comment associated with the node, entered when the node was cataloged. To change a comment in the node directory, uncatalog the node, and then catalog it again with the new comment.

Protocol

The communications protocol cataloged for the node.

Note: For information about fields associated with a specific node type, see the applicable CATALOG...NODE command.

Usage Notes

A node directory is created and maintained on each database client. It contains an entry for each remote workstation having databases that the client can access. The DB2 client uses the communication end point information in the node directory whenever a database connection or instance attachment is requested.

LIST NODE DIRECTORY

The database manager creates a node entry and adds it to the node directory each time it processes a CATALOG...NODE command. The entries can vary, depending on the communications protocol being used by the node.

The node directory can contain entries for the following types of nodes:

- APPC
- APPCLU
- APPN
- IPX/SPX
- Local
- Named pipe
- NetBIOS
- TCP/IP.

See Also

“CATALOG APPC NODE” on page 111
“CATALOG APPCLU NODE” on page 114
“CATALOG APPN NODE” on page 116
“CATALOG IPX/SPX NODE” on page 126
“CATALOG LOCAL NODE” on page 129
“CATALOG NAMED PIPE NODE” on page 131
“CATALOG NETBIOS NODE” on page 133
“CATALOG TCP/IP NODE” on page 136.

LIST NODEGROUPS

Lists all nodegroups associated with the current database.

Scope

This command can be issued from any node that is listed in `$HOME/sqllib/db2nodes.cfg`. It returns the same information from any of these nodes.

Authorization

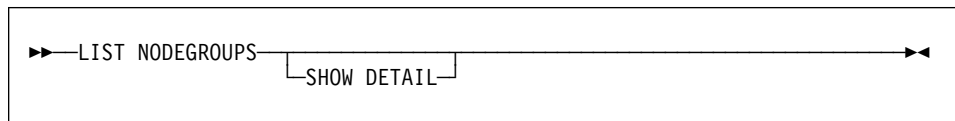
For the system catalogs `SYSCAT.NODEGROUPS` and `SYSCAT.NODEGROUPDEF`, one of the following is required:

- *sysadm* or *dbadm* authority
- CONTROL privilege
- SELECT privilege.

Required Connection

Database

Command Syntax



Command Parameters

SHOW DETAIL

Specifies that the output should include the following information:

- Partitioning map ID
- Node number
- In-use flag

Examples

Following is sample output from the LIST NODEGROUPS command:

```

  NODEGROUP NAME
  -----
  IBMCATGROUP
  IBMDEFAULTGROUP
  IBMTEMPGROUP

  3 record(s) selected.
  
```

LIST NODEGROUPS

Following is sample output from the LIST NODEGROUPS SHOW DETAIL command:

NODEGROUP NAME	PMAP_ID	NODE NUMBER	IN_USE
IBMCATGROUP	0		0 Y
IBMDEFAULTGROUP	1		0 Y

2 record(s) selected.

The fields are identified as follows:

NODEGROUP NAME

The name of the nodegroup. The name is repeated for each node in the nodegroup.

PMAP_ID

The ID of the partitioning map. The ID is repeated for each node in the nodegroup.

NODE NUMBER

The number of the node.

IN_USE

One of four values:

Y The node is being used by the nodegroup.

D The node is going to be dropped from the nodegroup as a result of a REDISTRIBUTE NODEGROUP operation. When the operation completes, the node will not be included in reports from LIST NODEGROUPS.

A The node has been added to the nodegroup but is not yet added to the partitioning map. The containers for the table spaces in the nodegroup have been added on this node. The value is changed to Y when the REDISTRIBUTE NODEGROUP operation completes successfully.

T The node has been added to the nodegroup, but is not yet added to the partitioning map. The containers for the table spaces in the nodegroup have not been added on this node. Table space containers must be added on the new node for each table space in the node group. The value is changed to A when containers have successfully been added.

For more information, see the *SQL Reference*.

See Also

“REDISTRIBUTE NODEGROUP” on page 337.

LIST NODES

Lists all nodes associated with the current database.

Scope

This command can be issued from any node that is listed in `$HOME/sqllib/db2nodes.cfg`. It returns the same information from any of these nodes.

Authorization

None

Required Connection

Database

Command Syntax

```
▶▶—LIST NODES—◀◀
```

Command Parameters

None

Examples

Following is sample output from the LIST NODES command:

```
NODE NUMBER
-----
          0
          2
          5
          7
          9

5 record(s) selected.
```

See Also

“REDISTRIBUTE NODEGROUP” on page 337.

LIST ODBC DATA SOURCES

LIST ODBC DATA SOURCES

Lists all available user or system ODBC data sources.

A *data source*, in ODBC (Open Database Connectivity) terminology, is a user-defined name for a specific database or file system. That name is used to access the database or file system through ODBC APIs. On Windows NT and Windows 95, either user or system data sources can be cataloged. A user data source is only visible to the user who cataloged it, whereas a system data source is visible to and can be used by all other users.

This command is available on Windows NT, Windows 95, and Windows 3.1 only.

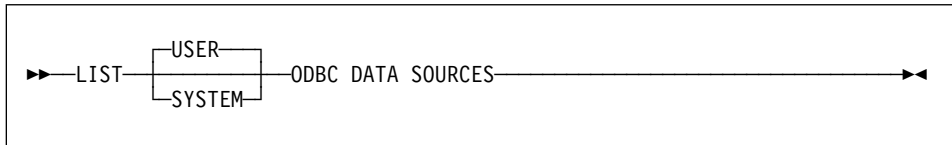
Authorization

None

Required Connection

None

Command Syntax



Command Parameters

USER

List only user ODBC data sources. This is the default if no keyword is specified.

SYSTEM

List only system ODBC data sources.

Example

The following is sample output from the LIST ODBC DATA SOURCES command:

User ODBC Data Sources	
Data source name	Description
-----	-----
SAMPLE	IBM DB2 ODBC DRIVER

LIST ODBC DATA SOURCES

See Also

“CATALOG ODBC DATA SOURCE” on page 135

“UNCATALOG ODBC DATA SOURCE” on page 404.

LIST PACKAGES/TABLES

LIST PACKAGES/TABLES

Lists packages or tables associated with the current database.

Authorization

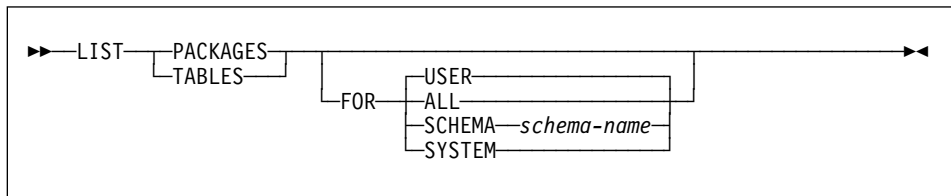
For the system catalogs SYSCAT.PACKAGES (LIST PACKAGES) and SYSCAT.TABLES (LIST TABLES), one of the following is required:

- *sysadm* or *dbadm* authority
- CONTROL privilege
- SELECT privilege.

Required Connection

Database. If implicit connect is enabled, a connection to the default database is established.

Command Syntax



Command Parameters

FOR

If the FOR clause is not specified, the packages or tables for USER are listed.

ALL

Lists all packages or tables in the database.

SCHEMA

Lists all packages or tables in the database for the specified schema only.

SYSTEM

Lists all system packages or tables in the database.

USER

Lists all user packages or tables in the database for the current user.

LIST PACKAGES/TABLES

Examples

The following is sample output from LIST PACKAGES:

Package	Schema	Bound by	Total sections	Valid	Format	Isolation level	Blocking
P1	SMITH	SMITH		1 Yes	0	CS	U

1 record(s) selected.

The following is sample output from LIST TABLES:

Table/View	Schema	Type	Creation time
DEPARTMENT	SMITH	T	1997-02-19-13.32.25.971890
EMP_ACT	SMITH	T	1997-02-19-13.32.27.851115
EMP_PHOTO	SMITH	T	1997-02-19-13.32.29.953624
EMP_RESUME	SMITH	T	1997-02-19-13.32.37.837433
EMPLOYEE	SMITH	T	1997-02-19-13.32.26.348245
ORG	SMITH	T	1997-02-19-13.32.24.478021
PROJECT	SMITH	T	1997-02-19-13.32.29.300304
SALES	SMITH	T	1997-02-19-13.32.42.973739
STAFF	SMITH	T	1997-02-19-13.32.25.156337

9 record(s) selected.

Usage Notes

LIST PACKAGES and LIST TABLES commands are available to provide a quick Version 1 interface to the system tables. However, Version 2 system tables offer a greater granularity of information, and should be used whenever possible.

The following Version 2 SELECT statements return similar information. They can be expanded to select the additional information that Version 2 provides.

```
select tabname, tabschema, type, create_time
from syscat.tables
order by tabschema, tabname;
```

```
select pkgname, pkgschema, boundby, total_sect,
       valid, format, isolation, blocking
from syscat.packages
order by pkgschema, pkgname;
```

```
select tabname, tabschema, type, create_time
from syscat.tables
where tabschema = 'SYSCAT'
order by tabschema, tabname;
```

LIST PACKAGES/TABLES

```
select pkgname, pkgschema, boundby, total_sect,  
       valid, format, isolation, blocking  
from syscat.packages  
where pkgschema = 'NULLID'  
order by pkgschema, pkgname;
```

```
select tabname, tabschema, type, create_time  
from syscat.tables  
where tabschema = USER  
order by tabschema, tabname;
```

```
select pkgname, pkgschema, boundby, total_sect,  
       valid, format, isolation, blocking  
from syscat.packages  
where pkgschema = USER  
order by pkgschema, pkgname;
```

LIST TABLESPACE CONTAINERS

Lists containers for the specified table space.

Scope

This command returns information only for the node on which it is executed.

Authorization

One of the following:

sysadm
sysctrl
sysmaint
dbadm

Required Connection

Database

Command Syntax

```
▶▶—LIST TABLESPACE CONTAINERS FOR—tablespace-id—▶▶  
└─SHOW DETAIL─┘
```

Command Parameters

FOR *tablespace-id*

An integer that uniquely represents a table space used by the current database. To get a list of all the table spaces used by the current database, use “LIST TABLESPACES” on page 271.

SHOW DETAIL

If this option is not specified, only the following basic information about each container is provided:

- Container ID
- Name
- Type (file, disk, or path).

If this option is specified, the following additional information about each container is provided:

- Total number of pages
- Number of useable pages
- Accessible (yes or no).

LIST TABLESPACE CONTAINERS

Examples

The following is sample output from LIST TABLESPACE CONTAINERS:

```
Tablespace Containers for Tablespace 0
Container ID          = 0
Name                 = /home/smith/smith/NODE0000/SQL00001/SQLT0000.0
Type                 = Path
```

The following is sample output from LIST TABLESPACE CONTAINERS with SHOW
DETAIL specified:

```
Tablespace Containers for Tablespace 0
Container ID          = 0
Name                 = /home/smith/smith/NODE0000/SQL00001/SQLT0000.0
Type                 = Path
Total pages          = 895
Useable pages        = 895
Accessible            = Yes
```

See Also

“LIST TABLESPACES” on page 271.

LIST TABLESPACES

Lists table spaces for the current database.

Scope

This command returns information only for the node on which it is executed.

Authorization

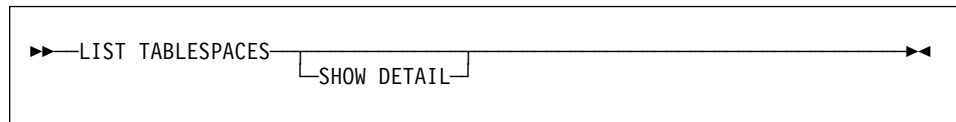
One of the following:

sysadm
sysctrl
sysmaint
dbadm

Required Connection

Database

Command Syntax



Command Parameters

SHOW DETAIL

If this option is not specified, only the following basic information about each table space is provided:

- Table space ID
- Name
- Type (system managed space or database managed space)
- Contents (any data, long data only, or temporary data)
- State, 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. "db2tbst - Get Tablespace State" on page 59 can be used to obtain the table space state associated with a given hexadecimal value. Following are the bit definitions listed in `sqlutil.h`:

0x0	Normal
0x1	Quiesced: SHARE
0x2	Quiesced: UPDATE
0x4	Quiesced: EXCLUSIVE
0x8	Load pending
0x10	Delete pending

LIST TABLESPACES

0x20	Backup pending
0x40	Roll forward in progress
0x80	Roll forward pending
0x100	Restore pending
0x100	Recovery pending (not used)
0x200	Disable pending
0x400	Reorg in progress
0x800	Backup in progress
0x1000	storage must be defined
0x2000	Restore in progress
0x2000000	storage may be defined
0x4000000	storDef is in 'final' state
0x8000000	storDef was changed prior to rollforward
0x10000000	dms rebalancer is active
0x20000000	TBS deletion in progress
0x40000000	TBS creation in progress
0x8	For service use only

If this option is specified, the following additional information about each table space is provided:

- Total number of pages
- Number of useable pages
- Number of used pages
- Number of free pages
- High water mark (in pages)
- Page size (in bytes)
- Extent size (in pages)
- Prefetch size (in pages)
- Number of containers
- Minimum recovery time (displayed only if not zero)
- State change table space ID (displayed only if the table space state is "load pending" or "delete pending")
- State change object ID (displayed only if the table space state is "load pending" or "delete pending")
- Number of quiescers (displayed only if the table space state is "quiesced: SHARE", "quiesced: UPDATE", or "quiesced: EXCLUSIVE")
- Table space ID and object ID for each quiescer (displayed only if the number of quiescers is greater than zero).

Examples

The following are two sample outputs from LIST TABLESPACES SHOW DETAIL.

```
Tablespaces for Current Database

Tablespace ID          = 0
Name                   = SYSCATSPACE
Type                   = System managed space
Contents               = Any data
State                  = 0x0000
  Detailed explanation:
    Normal
Total pages            = 895
```

LIST TABLESPACES

```

Useable pages                = 895
Used pages                   = 895
Free pages                   = Not applicable
High water mark (pages)     = Not applicable
Page size (bytes)           = 4096
Extent size (pages)         = 32
Prefetch size (pages)       = 32
Number of containers         = 1

Tablespace ID                = 1
Name                         = TEMPSPACE1
Type                         = System managed space
Contents                     = Temporary data
State                        = 0x0000
  Detailed explanation:
    Normal
Total pages                  = 1
Useable pages               = 1
Used pages                  = 1
Free pages                  = Not applicable
High water mark (pages)     = Not applicable
Page size (bytes)           = 4096
Extent size (pages)         = 32
Prefetch size (pages)       = 32
Number of containers         = 1

Tablespace ID                = 2
Name                         = USERSPACE1
Type                         = System managed space
Contents                     = Any data
State                        = 0x000c
  Detailed explanation:
    Quiesced: EXCLUSIVE
    Load pending
Total pages                  = 337
Useable pages               = 337
Used pages                  = 337
Free pages                  = Not applicable
High water mark (pages)     = Not applicable
Page size (bytes)           = 4096
Extent size (pages)         = 32
Prefetch size (pages)       = 32
Number of containers         = 1
State change tablespace ID   = 2
State change object ID       = 3
Number of quiescers          = 1
  Quiescer 1:
    Tablespace ID            = 2
    Object ID                 = 3

```

DB21011I In a partitioned database server environment, only the table spaces on the current node are listed.

Tablespaces for Current Database

```

Tablespace ID                = 0
Name                         = SYSCATSPACE
Type                         = System managed space
Contents                     = Any data

```

LIST TABLESPACES

```

State = 0x0000
  Detailed explanation:
    Normal
Total pages = 1200
Useable pages = 1200
Used pages = 1200
Free pages = Not applicable
High water mark (pages) = Not applicable
Page size (bytes) = 4096
Extent size (pages) = 32
Prefetch size (pages) = 32
Number of containers = 1

Tablespace ID = 1
Name = TEMPSPACE1
Type = System managed space
Contents = Temporary data
State = 0x0000
  Detailed explanation:
    Normal
Total pages = 1
Useable pages = 1
Used pages = 1
Free pages = Not applicable
High water mark (pages) = Not applicable
Page size (bytes) = 4096
Extent size (pages) = 32
Prefetch size (pages) = 32
Number of containers = 1

Tablespace ID = 2
Name = USERSPACE1
Type = System managed space
Contents = Any data
State = 0x0000
  Detailed explanation:
    Normal
Total pages = 1
Useable pages = 1
Used pages = 1
Free pages = Not applicable
High water mark (pages) = Not applicable
Page size (bytes) = 4096
Extent size (pages) = 32
Prefetch size (pages) = 32
Number of containers = 1

Tablespace ID = 3
Name = DMS8K
Type = Database managed space
Contents = Any data
State = 0x0000
  Detailed explanation:
    Normal
Total pages = 2000
Useable pages = 1952
Used pages = 96
Free pages = 1856
High water mark (pages) = 96

```

LIST TABLESPACES

```
Page size (bytes)           = 8192
Extent size (pages)        = 32
Prefetch size (pages)     = 32
Number of containers       = 2

Tablespace ID              = 4
Name                      = TEMP8K
Type                      = System managed space
Contents                  = Temporary data
State                     = 0x0000
  Detailed explanation:
    Normal
Total pages                = 1
Useable pages             = 1
Used pages                 = 1
Free pages                 = Not applicable
High water mark (pages)   = Not applicable
Page size (bytes)        = 8192
Extent size (pages)      = 32
Prefetch size (pages)    = 32
Number of containers      = 1
```

DB21011I In a partitioned database server environment, only the table spaces on the current node are listed.

Usage Notes

In a multi-node environment, this command does not return all the table spaces in the database. To obtain a list of all the table spaces, query SYSCAT.SYSTABLESPACES.

During a table space rebalance, the number of useable pages will include pages for the newly added container, but these new pages will 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 will equal the number of useable pages.

See Also

“db2tbst - Get Tablespace State” on page 59
“LIST TABLESPACE CONTAINERS” on page 269.

LOAD

LOAD

Loads data from files, tapes, or named pipes into a DB2 table. The load utility does not support loading data at the hierarchy level.

Scope

This command only affects the node on which it is executed.

In a multi-node environment, this command can be used only with ASC or DEL files. IXF files can be loaded only if the table exists on a single node nodegroup.

Authorization

One of the following:

sysadm
dbadm

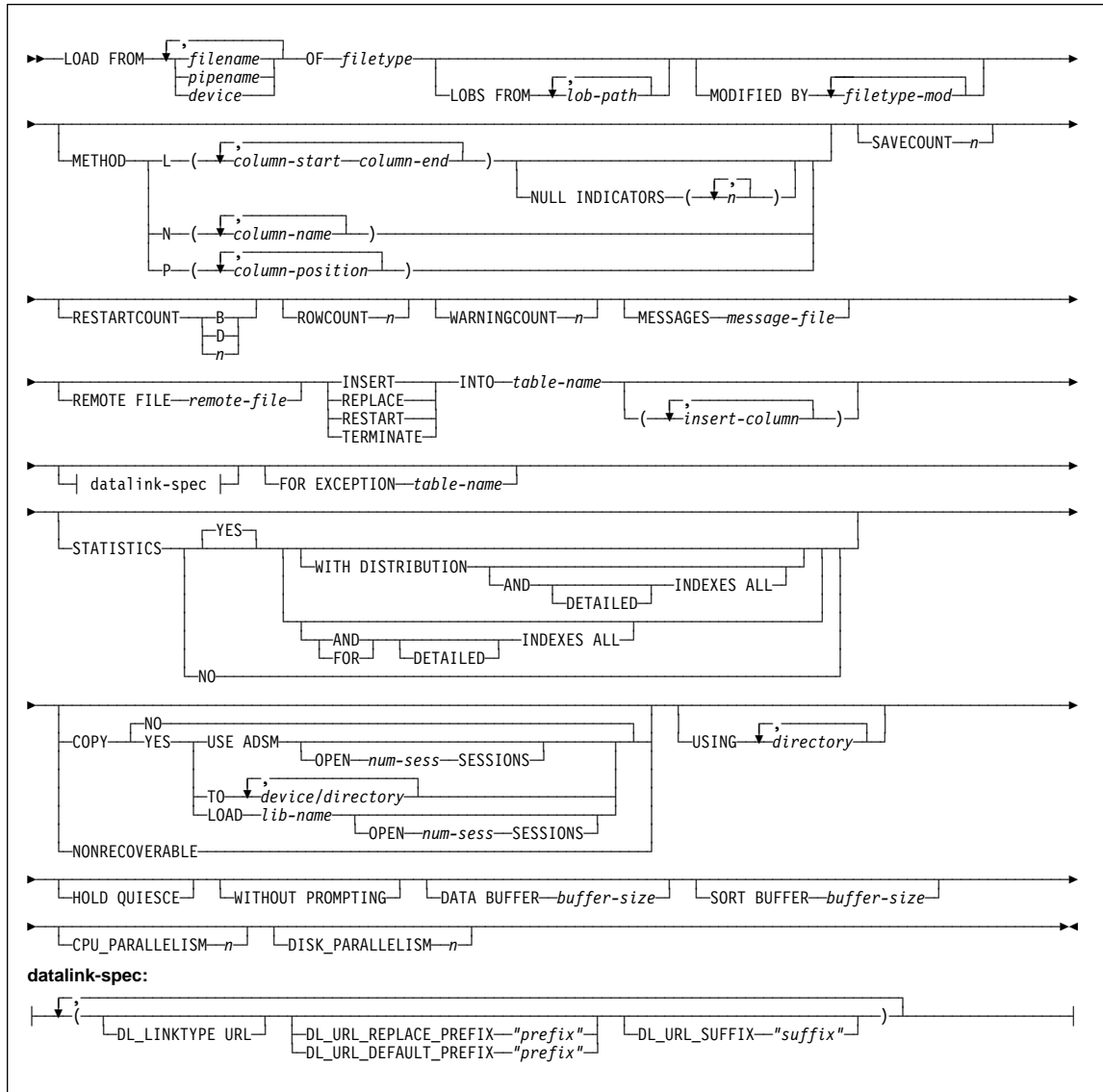
Note: Since all load processes (and all DB2 server processes, in general), are owned by the instance owner, and all of these processes use the identification of the instance owner to access needed files, the instance owner must have read access to input data files. These input data files must be readable by the instance owner, regardless of who invokes the command.

Required Connection

Database. If implicit connect is enabled, a connection to the default database is established.

Instance. An explicit attachment is not required. If a connection to the database has been established, an implicit attachment to the local instance is attempted.

Command Syntax



Command Parameters

FROM filename/pipename/device

Specifies the file, pipe, or device that contains the data being loaded. This file/pipe/device must reside on the node where the database resides. If several names are specified, they will be processed in sequence. If the

LOAD

last item specified is a tape device, the user is prompted for another tape. Valid response options are:

- c** Continue. Continue using the device that generated the warning message (for example, when a new tape has been mounted)
- d** Device terminate. Stop using the device that generated the warning message (for example, when there are no more tapes)
- t** Terminate. Terminate all devices.

Notes:

1. It is recommended that the fully qualified file name be used. If the server is remote, the fully qualified file name must be used. If the database resides on the same node as the caller, relative paths may be used.
2. Loading data from multiple IXF files is supported if the files are physically separate, but logically one file. It is *not* supported if the files are both logically and physically separate.
3. If, when specifying *pipename* on OS/2, less than the expected amount of data is loaded, clean up system resources (IPL is recommended), and reissue the LOAD command.

OF *filetype*

Specifies the format of the data in the input file:

- ASC (non-delimited ASCII format)
- DEL (delimited ASCII format)
- IXF (integrated exchange format, PC version), exported from the same or from another DB2 table.

For more information about file formats, see the Appendix C, “IMPORT/EXPORT/LOAD Utility File Formats” on page 433.

LOBS FROM *lob-path*

The path to the data files containing LOB values to be loaded. The path must end with a slash (/). The names of the LOB data files are stored in the main data file (ASC, DEL, or IXF), in the column that will be loaded into the LOB column. This option is ignored if *lobsinfile* is not specified within the *filetype-mod* string.

MODIFIED BY *filetype-mod*

Specifies additional information unique to the ASC, DEL, or IXF file format (see page 293).

METHOD

L

Specifies the start and end column numbers from which to load data.

Note: This method can only be used with ASC files, and is the only valid option for that file type.

N

Specifies the names of the columns in the data file to be loaded. The case of these column names must match the case of the corresponding names in the system catalogs. Each column in the table that is not nullable should be included in this list. Specify only complete subsets of column names (for example, given file columns F1, F2, F3, F4, F5, and F6, and table columns C1 INT, C2 INT NOT NULL, C3 INT NOT NULL, and C4 INT, method N (F1,F2,F3,F4) insert into table_name (C1,C2,C3,C4) is a valid request, while method N (F1,F4) is not valid, since there will be no data to put into C3.

Note: This method can only be used with IXF files.

P

Specifies the numbers of the columns to be loaded. Each column in the table that is not nullable should be included in this list. Specify only complete subsets of column numbers (for example, given file columns F1, F2, F3, F4, F5, and F6, and table columns C1 INT, C2 INT NOT NULL, C3 INT NOT NULL, and C4 INT, method P (1,2,3,4) is a valid request, while method P (1,4) is not valid.

Note: This method can only be used with IXF or DEL files, and is the only valid option for the DEL file type.

NULL INDICATORS *n*

Specifies a column (by number) to be used as a NULL indicator field. If this option is used, a NULL indicator column for each data column must also be specified. A value of zero indicates that the data column is not nullable, and that there will always be data in that column.

A value of Y in the NULL indicator column specifies that the column data is NULL. Any character *other than* Y in the NULL indicator column specifies that the column data is not NULL, and that column data specified by the METHOD L option will be loaded.

The NULL indicator character can be changed using the MODIFIED BY option (see page 293).

SAVECOUNT *n*

Specifies that LOAD is to establish consistency points after every *n* rows. This value is converted to a page count, and rounded up to intervals of the extent size. Since a message is issued at each consistency point, this option should be selected if the load will be monitored using "LOAD QUERY" on page 301. If the value of *n* is not sufficiently high, the synchronization of activities performed at each consistency point will impact performance.

The default value is 0, meaning that no consistency points will be established, unless necessary.

RESTARTCOUNT

LOAD

- B* LOAD will restart at the build phase.
- D* LOAD will restart at the delete phase.
- n* An integer specifying that the load is to be started at record *n*+1. The first *n* records are skipped.

This option can be specified with any of the INSERT, REPLACE, or RESTART modes. B or D must not be specified for the INSERT or the REPLACE mode.

Note: It may be necessary to quiesce the table spaces prior to invoking a load restart. To do this, issue:

```
db2 quiesce tablespaces for table tablename exclusive
```

ROWCOUNT *n*

Specifies the number of *n* physical records in the file to be loaded. Allows a user to load only the first *n* rows in a file.

WARNINGCOUNT *n*

Stops the load after *n* warnings. Set this parameter if no warnings are expected, but verification that the correct file and table are being used is desired. If *n* is 0, or this option is not specified, the load will continue regardless of the number of warnings issued.

If the load is stopped because the threshold of warnings was encountered, another load can be started in RESTART mode by specifying the RESTARTCOUNT option. Alternatively, another load can be initiated in REPLACE mode, starting at the beginning of the input file.

MESSAGES *message-file*

Specifies the destination for warning and error messages that occur during the load. If a message file is not specified, messages are written to a file in the current directory.

If the complete path to the file is not specified, LOAD uses the current directory and the default drive as the destination.

If the name of a file that already exists is specified, LOAD appends the information.

REMOTE FILE *remote-file*

Specifies the base name to be used when creating temporary files during a load, and should be fully qualified according to the server node.

These temporary files include a .log (load crash recovery log) file, a .rid file (containing a list of row identifiers for records that will be deleted during the delete phase), a .dlr file (containing a list of row identifiers for records that will be deleted due to DATALINK violations), and a binary .msg file (containing load messages). None of these files are viewed directly by the user. The size of each depends on a number of variables:

- The .log file size is usually a few hundred bytes, unless there are LF or LOB values on the table, in which case this file also contains a mirror of the allocation pages. The allocation pages for LOBs can consume a significant amount of space.
- The .rid and .dlr files are normally zero bytes in length. However, if there are many records to delete, these files can grow substantially (approximately four bytes for each record to be deleted).
- The .msg file size is usually a few hundred bytes, but it increases with the number of consistency points and warnings. This file is a compressed, binary form of all the SQL message information returned once the load operation has completed.

For more information about remote files, see page 299.

INSERT

One of four modes under which the load utility can execute. Adds the loaded data to the table without changing the existing table data.

REPLACE

One of four modes under which the load utility can execute. Deletes all existing data from the table, and inserts the loaded data. The table definition and index definitions are not changed.

RESTART

One of four modes under which the load utility can execute. Restarts LOAD after a previous load was interrupted.

It is important to keep track of the last commit point. This information is stored in the message file and is passed to LOAD. Use "LOAD QUERY" on page 301 to get this information if the database connection was lost during the load.

TERMINATE

One of four modes under which the load utility can execute. Terminates a previously interrupted load and moves the table spaces in which the table resides from load pending state to recovery pending state. The table spaces cannot be used until a backup has been restored and the table spaces have been rolled forward. A restart should be issued before attempting to complete an interrupted load.

Notes:

1. This option is not recommended for general use; it should only be selected if an unrecoverable error has occurred.
2. Table spaces must be explicitly quiesced in exclusive mode before invoking this option.
3. To terminate a load operation, the LOAD command (with all parameters originally specified) must be issued with the keyword TERMINATE replacing the mode originally specified.

INTO *table-name*

Specifies the database table into which the data is to be loaded. This table cannot be a system table. An alias, or the fully qualified or unqualified table

LOAD

name can be specified. A qualified table name is in the form *schema.tablename*. If an unqualified table name is specified, the table will be qualified with the current authorization ID.

insert-column

Specifies the table column into which the data is to be inserted.

The load utility cannot parse columns whose names contain one or more spaces. For example,

```
db2 load from delfile1 of del modified by noeofchar noheader
method P (1, 2, 3, 4, 5, 6, 7, 8, 9)
insert into table1 (BLOB1, S2, I3, Int 4, I5, I6, DT7, I8, TM9)
```

will fail because of the Int 4 column. The solution is to enclose such column names with double quotation marks:

```
db2 load from delfile1 of del modified by noeofchar noheader
method P (1, 2, 3, 4, 5, 6, 7, 8, 9)
insert into table1 (BLOB1, S2, I3, "Int 4", I5, I6, DT7, I8, TM9)
```

FOR EXCEPTION *table-name*

Specifies the exception table into which rows in error will be copied. Any row that is in violation of a unique index or a primary key index is copied.

Information that is written to the exception table is *not* written to the dump file (for a description of the `dumpfile` modifier, see Table 8 on page 293). The exception table contains rows that are in violation of a unique index or a primary key index. In a partitioned database environment, an exception table must be defined for those nodes on which the loading table is defined. The dump file, on the other hand, contains rows that cannot be loaded because they are invalid or have syntax errors.

STATISTICS YES

Specifies that statistics will be gathered for the table and for any existing indexes. This option is not supported if the load is in INSERT or in RESTART mode.

WITH DISTRIBUTION

Specifies that distribution statistics are requested.

AND INDEXES ALL

Update statistics for both the table and its indexes.

FOR INDEXES ALL

Update statistics for the indexes only.

DETAILED

Specifies that extended index statistics are requested.

STATISTICS NO

Specifies that no statistics will be gathered, and that the statistics in the catalogs will not be altered.

COPY NO

Specifies that the table space in which the table resides will be placed in backup pending state if forward recovery is enabled (that is, *logretain* or *userexit* is on). The data will not be accessible until a table space backup or a full database backup is made.

COPY YES

Specifies that a copy of the loaded data will be saved. This option is invalid if forward recovery is disabled (both *logretain* and *userexit* are off).

USE ADISM

Specifies that the copy will be stored using ADISM.

OPEN *num-sess* SESSIONS

The number of I/O sessions to be used with ADISM or the vendor product. The default value is 1.

TO *device/directory*

Specifies the device or directory on which the copy image will be created.

LOAD *lib-name*

The name of the shared library (DLL on OS/2 or the Windows operating system) containing the vendor backup and restore I/O functions to be used. It may contain the full path. If the full path is not given, it will default to the path where the user exit programs reside.

NONRECOVERABLE

Specifies that the load transaction is to be marked as non-recoverable, and that it will not be possible to recover it by a subsequent rollforward action. The rollforward utility will skip the transaction, and will mark the table into which data was being loaded as "invalid". The utility will also ignore any subsequent transactions against that table. After the roll forward is completed, such a table can only be dropped.

With this option, table spaces are not put in backup pending state following the load operation, and a copy of the loaded data does not have to be made during the load.

USING *directory*

Specifies the directory that holds the temporary files needed for index creation. The default value is */sqlib/tmp/*. In a partitioned database environment, load operations occurring simultaneously on different nodes may erode performance, because each operation will try to write to this default directory. To avoid this scenario, specify a directory local to the machine from which LOAD is being invoked.

Note: The size of this directory should be at least 130% of the index size.

HOLD QUIESCE

Specifies that the utility should leave the table in quiesced exclusive state after the load operation. To quiesce the table spaces, issue:

```
db2 quiesce tablespaces for table tablename reset
```

Note: Ensure that no *phantom quiesces* are created (see "QUIESCE TABLESPACES FOR TABLE" on page 328).

WITHOUT PROMPTING

Specifies that the list of data files contains all the files that are to be loaded, and that the devices/directories listed are sufficient for the entire load. If a continuation input file is not found, or the copy targets are filled

LOAD

before the load finishes, the load will fail, and the table will remain in load pending state.

If this option is not specified, and the tape device encounters an end of tape for the copy image, or the last item listed is a tape device, the user is prompted for a new tape on that device.

DATA BUFFER *buffer-size*

Specifies the number of 4KB pages (regardless of the degree of parallelism) to use as buffered space for transferring data within the utility. If the value specified is less than the algorithmic minimum, the minimum required resource is used, and no warning is returned.

This memory is allocated directly from the utility heap, whose size can be modified through the *util_heap_sz* database configuration parameter.

If a value is not specified, an intelligent default is calculated by the utility at run time. The default is based on a percentage of the free space available in the utility heap at the instantiation time of the loader, as well as some characteristics of the table.

SORT BUFFER *buffer-size*

Use this parameter to specify the number of 4KB pages of memory that are to be used for sorting the index keys during a load operation.

Note: Sort buffer size has a very large impact on sort performance. Therefore, for very large tables (for example, tables in excess of 100M), this buffer should be set as large as possible.

If a value is not specified, the utility uses the larger of:

- 2MB for OS/2 or Windows NT, or 6MB for all other platforms
- The minimum size allowed by the sort algorithm
- 15% of the free space remaining in the utility heap.

If a value greater than zero, but less than the required minimum is specified, the minimum value for that load is used.

CPU_PARALLELISM *n*

Specifies the number of processes or threads that the load utility will spawn for parsing, converting, and formatting records when building table objects. This parameter is designed to exploit SMP parallelism. It is particularly useful when loading presorted data, because record order in the source data is preserved. If the value of this parameter is zero, the load utility uses an intelligent default value at run time.

Notes:

1. If this parameter is used with tables containing either LOB or LONG VARCHAR fields, its value becomes one, regardless of the number of system CPUs or the value specified by the user.
2. Specifying a small value for the *savecount* parameter causes the loader to perform many more IO operations to flush both data and table metadata. When *cpu_parallelism* is greater than one, the flushing operations are asynchronous, permitting the loader to exploit the CPU. When *cpu_parallelism* is set to one, the loader waits on IO during

consistency points. A load operation with *cpu_parallelism* set to two, and *savecount* set to one, completes faster than the same load performed with *cpu_parallelism* set to one, even though there is only one CPU.

DISK_PARALLELISM *n*

Specifies the number of processes or threads that the load utility will spawn for writing data to the table space containers. If a value is not specified, the utility selects an intelligent default based on the number of table space containers and the characteristics of the table.

DL_LINKTYPE

If specified, it should match the LINKTYPE of the column definition. Thus, DL_LINKTYPE URL is acceptable if the column definition specifies LINKTYPE URL.

DL_URL_DEFAULT_PREFIX "*prefix*"

If specified, it should act as the default prefix for all DATALINK values within the same column. In this context, prefix refers to the "scheme host port" part of the URL specification.

Examples of prefix are:

```
"http://server"
"file://server"
"file:"
"http://server:80"
```

If no prefix is found in a column's data, and a default prefix is specified with DL_URL_DEFAULT_PREFIX, the default prefix is prefixed to the column value (if not NULL).

For example, if DL_URL_DEFAULT_PREFIX specifies the default prefix "http://toronto":

- The column input value "/x/y/z" is stored as "http://toronto/x/y/z".
- The column input value "http://coyote/a/b/c" is stored as "http://coyote/a/b/c".
- The column input value NULL is stored as NULL.

DL_URL_REPLACE_PREFIX "*prefix*"

This clause is useful while loading or importing data previously generated by the export utility, when the user wants to globally replace the host name in the data with another host name. If specified, it becomes the prefix for *all* non-NULL column values. If a column value has a prefix, this will replace it. If a column value has no prefix, the prefix specified by DL_URL_REPLACE_PREFIX is prefixed to the column value.

For example, if DL_URL_REPLACE_PREFIX specifies the prefix "http://toronto":

- The column input value "/x/y/z" is stored as "http://toronto/x/y/z".
- The column input value "http://coyote/a/b/c" is stored as "http://toronto/a/b/c". Note that "toronto" replaces "coyote".

LOAD

- The column input value NULL is stored as NULL.

DL_URL_SUFFIX *"suffix"*

If specified, it is appended to every non-NULL column value for the column. It is, in fact, appended to the "path" component of the URL part of the DATALINK value.

Examples

Example 1

TABLE1 has 5 columns:

```
COL1 VARCHAR 20 NOT NULL WITH DEFAULT
COL2 SMALLINT
COL3 CHAR 4
COL4 CHAR 2 NOT NULL WITH DEFAULT
COL5 CHAR 2 NOT NULL
```

ASCFILE1 has 6 elements:

```
ELE1 positions 01 to 20
ELE2 positions 21 to 22
ELE5 positions 23 to 23
ELE3 positions 24 to 27
ELE4 positions 28 to 31
ELE6 positions 32 to 32
ELE6 positions 33 to 40
```

Data Records:

```
1...5...10...15...20...25...30...35...40
Test data 1      XXN 123abcdN
Test data 2 and 3  QQY  wxyzN
Test data 4,5 and 6 WVN6789  Y
```

The following command loads the table from the file:

```
db2 load from ascfile1 of asc modified by striptblanks reclen=40
method L (1 20, 21 22, 24 27, 28 31)
null indicators (0,0,23,32)
insert into table1 (col1, col5, col2, col3)
```

Notes:

1. The specification of `striptblanks` in the `MODIFIED BY` parameter forces the truncation of blanks in `VARCHAR` columns (`COL1`, for example, which is 11, 17 and 19 bytes long, in rows 1, 2 and 3, respectively).
2. The specification of `reclen=40` in the `MODIFIED BY` parameter indicates that there is no new-line character at the end of each input record, and that each record is 40 bytes long. The last 8 bytes are not used to load the table.
3. Since `COL4` is not provided in the input file, it will be inserted into `TABLE1` with its default value (it is defined `NOT NULL WITH DEFAULT`).

4. Positions 23 and 32 are used to indicate whether COL2 and COL3 of TABLE1 will be loaded NULL for a given row. If there is a Y in the column's null indicator position for a given record, the column will be NULL. If there is an N, the data values in the column's data positions of the input record (as defined in L(.....)) are used as the source of column data for the row. In this example, neither column in row 1 is NULL; COL2 in row 2 is NULL; and COL3 in row 3 is NULL.
5. In this example, the NULL INDICATORS for COL1 and COL5 are specified as 0 (zero), indicating that the data is not nullable.
6. The NULL INDICATOR for a given column can be anywhere in the input record, but the position must be specified, and the Y or N values must be supplied.

Example 2 (Loading LOBs from Files)

TABLE1 has 3 columns:

```
COL1 CHAR 4 NOT NULL WITH DEFAULT
LOB1 LOB
LOB2 LOB
```

ASCFILE1 has 3 elements:

```
ELE1 positions 01 to 04
ELE2 positions 06 to 13
ELE3 positions 15 to 22
```

The following files reside in either /u/user1 or /u/user1/bin:

```
ASCFILE2 has LOB data
ASCFILE3 has LOB data
ASCFILE4 has LOB data
ASCFILE5 has LOB data
ASCFILE6 has LOB data
ASCFILE7 has LOB data
```

Data Records in ASCFILE1:

```
1...5...10...15...20...25...30.
REC1 ASCFILE2 ASCFILE3
REC2 ASCFILE4 ASCFILE5
REC3 ASCFILE6 ASCFILE7
```

The following command loads the table from the file:

```
db2 load from ascfile1 of asc
lobs from /u/user1, /u/user1/bin
modified by lobsinfile reqlen=22
method L (1 4, 6 13, 15 22)
insert into table1
```

LOAD

Notes:

1. The specification of `lobsinfile` in the `MODIFIED BY` parameter tells the loader that all LOB data is to be loaded from files.
2. The specification of `recLen=22` in the `MODIFIED BY` parameter indicates that there is no new-line character at the end of each input record, and that each record is 22 bytes long.
3. LOB data is contained in 6 files, `ASCFILE2` through `ASCFILE7`. Each file contains the data that will be used to load a LOB column for a specific row. The relationship between LOBs and other data is specified in `ASCFILE1`. The first record of this file tells the loader to place `REC1` in `COL1` of row 1. The contents of `ASCFILE2` will be used to load `LOB1` of row 1, and the contents of `ASCFILE3` will be used to load `LOB2` of row 1. Similarly, `ASCFILE4` and `ASCFILE5` will be used to load `LOB1` and `LOB2` of row 2, and `ASCFILE6` and `ASCFILE7` will be used to load the LOBs of row 3.
4. The `LOBS FROM` parameter contains 2 paths that will be searched for the named LOB files when those files are required by the loader.
5. To load LOBs directly from `ASCFILE1` (a non-delimited ASCII file), without the `LOBSINFILE` option, the following rules must be observed:
 - The total length of any record, including LOBs, cannot exceed 32K.
 - LOB fields in the input records must be of fixed length, and LOB data padded with blanks as necessary.
 - The `striptblanks` option of `MODIFIED BY` must be specified, so that the trailing blanks used to pad LOBs can be removed as the LOBs are inserted into the database.

Example 3 (Using Dump Files)

Table `FRIENDS` is defined as:

```
table friends "( c1 INT NOT NULL, c2 INT, c3 CHAR(8) )"
```

If an attempt is made to load the following data records into this table,

```
23, 24, bobby  
, 45, john  
4,, mary
```

the second row is rejected because the first `INT` is `NULL`, and the column definition specifies `NOT NULL`. Columns which contain initial characters that are not consistent with the `DEL` format will generate an error, and the record will be rejected. Such records can be written to a dump file (see Table 8 on page 293).

`DEL` data appearing in a column outside of character delimiters is ignored, but does generate a warning. For example:

```
22,34,"bob"  
24,55,"sam" sdf
```

The utility will load "sam" in the third column of the table, and the characters "sdf" will be flagged in a warning. The record is not rejected. Another example:

```
22 3, 34,"bob"
```

The utility will load 22,34,"bob", and generate a warning that some data in column one following the 22 was ignored. The record is not rejected.

Example 4 (Loading DATALINK Data)

The following command loads the table MOVIE TABLE from the input file delfile1, which has data in the DEL format:

```
db2 load from delfile1 of del
      modified by d1del|
      insert into movietable (actorname, description, url_making_of, url_movie)
      (d1_url_default_prefix "http://narang"),
      (d1_url_replace_prefix "http://bomdel" d1_url_suffix ".mpeg")
      for exception excptab
```

Notes:

1. The table has four columns:

actorname	VARCHAR(n)
description	VARCHAR(m)
url_making_of	DATALINK (with LINKTYPE URL)
url_movie	DATALINK (with LINKTYPE URL)

2. The DATALINK data in the input file has the vertical bar (|) character as the sub-field delimiter.
3. If any column value for url_making_of does not have the prefix character sequence, "http://narang" is used.
4. Each non-NULL column value for url_movie will get "http://bomdel" as its prefix. Existing values are replaced.
5. Each non-NULL column value for url_movie will get ".mpeg" appended to the path. For example, if a column value of url_movie is "http://server1/x/y/z", it will be stored as "http://bomdel/x/y/z.mpeg"; if the value is "/x/y/z", it will be stored as "http://bomdel/x/y/z.mpeg".
6. If any unique index or DATALINK exception occurs while loading the table, the affected records are deleted from the table and put into the exception table excptab.

Usage Notes

Data is loaded in the sequence that appears in the input file. If a particular sequence is desired, the data should be sorted before a load is attempted.

The load utility builds indexes based on existing definitions. The exception tables are used to handle duplicates on unique keys. The utility does not enforce referential integrity, perform constraints checking, or update summary tables that are dependent

LOAD

on the tables being loaded. Tables being loaded that include referential or check constraints are placed in check pending state. Summary tables dependent on tables being loaded are also placed in check pending state. Issue the SET CONSTRAINTS statement to take the tables out of check pending state. Load operations cannot be carried out on replicated summary tables.

If clustering is required, the data should be sorted on the clustering index prior to loading.

If multiple concurrent load operations are to be run, unique USING and REMOTE FILE values must be specified, otherwise temporary files created by the different load operations may conflict.

DB2 File Manager Considerations

For each DATALINK column, there can be one column specification within parentheses. Each column specification consists of one or more of DL_LINKTYPE, *prefix* and a DL_URL_SUFFIX specification. The *prefix* information can be either DL_URL_REPLACE_PREFIX, or the DL_URL_DEFAULT_PREFIX specification.

There can be as many DATALINK column specifications as the number of DATALINK columns defined in the table. The order of specifications follows the order of DATALINK columns as found within the insert-column list (if specified by INSERT INTO (insert-column, ...)), or within the table definition (if insert-column is not specified).

For example, if a table has columns C1, C2, C3, C4, and C5, and among them only columns C2 and C5 are of type DATALINK, and the insert-column list is (C1, C5, C3, C2), there should be two DATALINK column specifications. The first column specification will be for C5, and the second column specification will be for C2. If an insert-column list is not specified, the first column specification will be for C2, and the second column specification will be for C5.

If there are multiple DATALINK columns, and some columns do not need any particular specification, the column specification should have at least the parentheses to unambiguously identify the order of specifications. If there are no specifications for any of the columns, the entire list of empty parentheses can be dropped. Thus, in cases where the defaults are satisfactory, there need not be any DATALINK specification.

It is possible that while running the load utility, the connection between DB2 and the file server may fail. This would cause the load operation to fail. The user response to this problem is as follows:

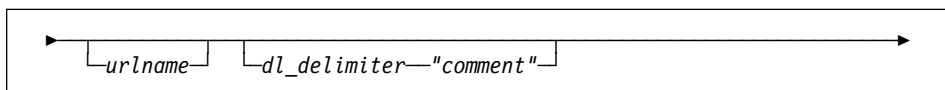
1. Start the file server and DB2 DB2 File Manager.
2. Restart the load utility. It will restart from the last consistency point.

Representation of DATALINK Information in an Input File

The LINKTYPE (currently only URL is supported) is not specified as part of DATALINK information. The LINKTYPE is specified in the LOAD or the IMPORT command, and for

input files of type PC/IXF, in the appropriate column descriptor records as described in the Appendix C, "IMPORT/EXPORT/LOAD Utility File Formats" on page 433.

The syntax of DATALINK information for a URL LINKTYPE is as follows:



Note that both *urlname* and *comment* are optional. If neither is provided, the null value is assigned.

urlname

The URL name must conform to valid URL syntax.

Notes:

1. Only "http" and "file" are permitted as a scheme name.
2. The prefix (scheme, host, and port) of the URL name is optional. If a prefix is not present, it is taken from the DL_URL_DEFAULT_PREFIX or the DL_URL_REPLACE_PREFIX specification of the load or the import utility. If none of these is specified, the prefix defaults to "file://localhost". Thus, in the case of local files, the file name with full path name can be entered as the URL name, without the need for a DATALINK column specification within the LOAD or the IMPORT command.
3. Prefixes, even if present in URL names, are overridden by a different prefix name on the DL_URL_REPLACE_PREFIX specification during a load or import operation.
4. The "path" (after appending DL_URL_SUFFIX, if specified) is the full path name of the remote file in the remote server. Relative path names are not allowed. The http server default path-prefix is not taken into account.

dl_delimiter

A character specified via `d1del`, or defaulted to on the LOAD or the IMPORT command. Whitespace characters (blanks, tabs, and so on) are permitted before and after the value specified for this parameter.

comment

If specified, the *comment* text must be enclosed by the character string delimiter, which is double quotation marks (") by default. Note that for delimited ASCII file formats (DEL), the character string delimiter can be overridden by the MODIFIED BY *filetype-mod* specification of the LOAD or the IMPORT command.

If no comment is specified, the comment defaults to a string of length zero.

Following are valid examples of data of type DATALINK (assume that the LOAD or IMPORT specification for the column is DL_URL_DEFAULT_PREFIX "http://qso"):

LOAD

- `http://www.almaden.ibm.com:80/mrep/intro.mpeg; "Intro Movie"`

This is stored with the following parts:

- `scheme = http`
- `server = www.almaden.ibm.com`
- `path = /mrep/intro.mpeg`
- `comment = "Intro Movie"`

- `file://narang/u/narang; "InderPal's Home Page"`

This is stored with the following parts:

- `scheme = file`
- `server = narang`
- `path = /u/narang`
- `comment = "InderPal's Home Page"`

- `file://narang/pics/xxx.jpeg?search_pat`

This is stored with the following parts:

- `scheme = file`
- `server = narang`
- `path = /pics/xxx.jpeg`
- `comment = NULL string`

- `/u/me/myfile.ps`

This is stored with the following parts:

- `scheme = http`
- `server = qso`
- `path = /u/me/myfile.ps`
- `comment = NULL string`

- `file:/home/ff.gg ; "hi there"`

This is stored with the following parts:

- `scheme = file`
- `server = localhost`
- `path = /home/ff.gg`
- `comment = "hi there"`

File Type Modifications

Note: The load utility does not issue a warning if an attempt is made to use unsupported file types with the MODIFIED BY option. If this is attempted, the load fails, and an error code is returned.

Table 8 (Page 1 of 7). Valid File Type Modifications (LOAD)

Modification	Description
All File Formats	
anyorder	This modifier is used in conjunction with the <i>cpu_parallelism</i> parameter. Specifies that the preservation of source data order is not required, yielding significant additional performance benefit on SMP systems. If the value of <i>cpu_parallelism</i> is 1, this option is ignored. This option is not supported if SAVECOUNT > 0, since crash recovery after a consistency point requires that data be loaded in sequence.
dldelx	<i>x</i> is a single character DATALINK delimiter. The default value is a semicolon (;). The specified character is used in place of a semicolon as the inter-field separator for a DATALINK value. It is needed because a DATALINK value may have more than one sub-value. <code>abc</code> Note: For DEL (delimited ASCII) files, <i>x</i> must not be the same character specified as the row, column, or character string delimiter.
fastparse	Reduced data checking is done on user-supplied column values, and performance is enhanced. Tables loaded under this option are guaranteed to be architecturally correct, and the utility is guaranteed to perform sufficient data checking to prevent a segmentation violation or trap. Data that is in correct form will be loaded correctly. This option does not affect referential integrity checking or constraints checking; it merely reduces syntax checking of the supplied data. For example, if the value 123qwr4 were encountered as a field entry for an integer column in an ASC file, the load utility would ordinarily flag a syntax error, since the value does not represent a valid number. With <i>fastparse</i> , a syntax error is not detected, and an arbitrary number is loaded into the integer field. Care must be taken to use this modifier with clean data only. Performance improvements using this option with ASCII data can be quite substantial, but <i>fastparse</i> does not significantly enhance performance with PC/IXF data, since IXF is a binary format, and <i>fastparse</i> affects parsing and conversion from ASCII to internal forms.

LOAD

Modification	Description
indexfreespace= <i>x</i>	<p><i>x</i> is an integer between 0 and 99 inclusive. The value is interpreted as the percentage of each index page that is to be left as free space when loading the index. The first entry in a page is added without restriction; subsequent entries are added if the percent free space threshold can be maintained. The default value is the one used at CREATE INDEX time.</p> <p>This value takes precedence over the PCTFREE value specified in the CREATE INDEX statement, and affects index leaf pages only.</p>
lobsinfile	<i>lob-path</i> specifies the path to the files containing LOB values. The ASC, DEL, or IXF load input files contain the names of the files having LOB data in the LOB column.
noheader	<p>Skips the header verification code.</p> <p>"db2split - Data Declustering Tool" on page 53 writes a header to each file contributing data to a table in a multi-node nodegroup. The header includes the node number, the partitioning map, and the partitioning key specification. The load utility requires this information to verify that the data is being loaded at the right node. When loading files into a table that exists on a single-node nodegroup, the headers do not exist, and this option causes the load utility to skip the header verification code.</p>
norowwarnings	Suppresses all warnings about rejected rows.
pagefreespace= <i>x</i>	<p><i>x</i> is an integer between 0 and 100 inclusive. The value is interpreted as the percentage of each data page that is to be left as free space.</p> <p>If the specified value is invalid because of the minimum row size, (for example, a row that is at least 3000 bytes long, and an <i>x</i> value of 50), the row will be placed on a new page. If a value of 100 is specified, each row will reside on a new page.</p> <p>Note: The PCTFREE value of a table determines the amount of free space designated per page. If a pagefreespace value on the load operation or a PCTFREE value on a table have not been set, the utility will fill up as much space as possible on each page. The value set by pagefreespace overrides the PCTFREE value specified for the table.</p>
totalfreespace= <i>x</i>	<p><i>x</i> is an integer between 0 and 100 inclusive. The value is interpreted as the percentage of the total pages in the table that is to be appended to the end of the table as free space. For example, if <i>x</i> is 20, and the table has 100 data pages, 20 additional empty pages will be appended. The total number of data pages for the table will be 120.</p>

Table 8 (Page 3 of 7). Valid File Type Modifications (LOAD)

Modification	Description
usedefaults	<p>If a source column for a target table column has been specified, but it contains no data for one or more row instances, default values are loaded. Examples of missing data are:</p> <ul style="list-style-type: none"> • For DEL files: ",," is specified for the column • For DEL/ASC/WSF files: A row that does not have enough columns, or is not long enough for the original specification. <p>Without this option, if a source column contains no data for a row instance, one of the following occurs:</p> <ul style="list-style-type: none"> • If the column is nullable, a NULL is loaded • If the column is not nullable, the utility rejects the row.
ASCII File Formats (ASC/DEL)	
codepage=x	<p>x is an ASCII character string. The value is interpreted as the code page of the data in the input data set. Converts character data (and numeric data specified in characters) from this code page to the database code page during the load operation.</p> <p>The following rules apply:</p> <ul style="list-style-type: none"> • For pure DBCS (graphic), mixed DBCS, and EUC, delimiters are restricted to the range of x00 to x3F, inclusive. • For DEL data specified in an EBCDIC code page, the delimiters may not coincide with the shift-in and shift-out DBCS characters. • nullindchar must specify symbols included in the standard ASCII set between code points x20 and x7F, inclusive. This refers to ASCII symbols and code points. EBCDIC data can use the corresponding symbols, even though the code points will be different.

LOAD

Table 8 (Page 4 of 7). Valid File Type Modifications (LOAD)

Modification	Description
dumpfile = x	<p>x is the fully qualified (according to the server node) name of an exception file to which rejected rows are written. A maximum of 32K of data is written per record. Following is an example that shows how to specify a dump file:</p> <pre>db2 load from data of del modified by dumpfile = /u/user/filename insert into table_name</pre> <p>Notes:</p> <ol style="list-style-type: none"> 1. In a partitioned database environment, the path should be local to the loading node, so that concurrently running load operations do not attempt to write to the same file. 2. The contents of the file are written to disk in an asynchronous buffered mode. In the event of a failed or an interrupted load operation, the number of records committed to disk cannot be known with certainty, and consistency cannot be guaranteed after a LOAD RESTART. The file can only be assumed to be complete for a load operation that starts and completes in a single pass. 3. This modifier does not support file names with multiple file extensions. For example, <pre>dumpfile = /home/svtdbm6/DUMP.FILE</pre> is acceptable to the load utility, but <pre>dumpfile = /home/svtdbm6/DUMP.LOAD.FILE</pre> is not.
implieddecimal	The location of an implied decimal point is determined by the column definition; it is no longer assumed to be at the end of the value. For example, the value 12345 is loaded into a DECIMAL(8,2) column as 123.45, <i>not</i> 12345.00.
noeofchar	The optional end-of-file character x'1A' is not recognized as the end of file. Processing continues as if it were a normal character.
ASC (Non-delimited ASCII) File Format	

Table 8 (Page 5 of 7). Valid File Type Modifications (LOAD)

Modification	Description
binarynumerics	<p>Numeric (but not DECIMAL) data must be in binary form, not the character representation. This avoids costly conversions.</p> <p>This option is supported only with positional ASC, using fixed length records specified by the reclen option. The noeofchar option is assumed.</p> <p>The following rules apply:</p> <ul style="list-style-type: none"> • No conversion between data types is performed, with the exception of BIGINT, INTEGER, and SMALLINT. • Data lengths must match their target column definitions. • FLOATs must be in IEEE Floating Point format. • Binary data in the load source file is assumed to be big-endian, regardless of the platform on which LOAD is running. <p>Note: NULLs cannot be present in the data for columns affected by this modifier. Blanks (normally interpreted as NULL) are interpreted as a binary value when this modifier is used.</p>
nullindchar=x	<p>x is a single character. Changes the character denoting a null value to x. The default value of x is Y.^b</p> <p>This modifier is case sensitive for EBCDIC data files, except when the character is an English letter. For example, if the null indicator character is specified to be the letter N, then n is also recognized as a null indicator.</p>
packeddecimal	<p>Loads packed-decimal data directly, since the binarynumerics modifier does not include the DECIMAL field type.</p> <p>This option is supported only with positional ASC, using fixed length records specified by the reclen option. The noeofchar option is assumed.</p> <p>Supported values for the sign nibble are:</p> <pre style="margin-left: 2em;">+ = 0xC 0xA 0xE 0xF - = 0xD 0xB</pre> <p>Note: NULLs cannot be present in the data for columns affected by this modifier. Blanks (normally interpreted as NULL) are interpreted as a binary value when this modifier is used.</p> <p>Regardless of the server platform, the byte order of binary data in the load source file is assumed to be big-endian; that is, when using this modifier on OS/2 or on the Windows operating system, the byte order must not be reversed.</p>
reclen=x	<p>x is an integer with a maximum value of 32767. x characters are read for each row, and a new-line character is not used to indicate the end of the row.</p>

LOAD

<i>Table 8 (Page 6 of 7). Valid File Type Modifications (LOAD)</i>	
Modification	Description
striptblanks	<p>Truncates any trailing blank spaces when loading data into a variable-length field. If this option is not specified, blank spaces are kept.</p> <p>This option cannot be specified together with striptnulls. These are mutually exclusive options.</p> <p>Note: This option replaces the obsolete t option, which is supported for back-level compatibility only.</p>
striptnulls	<p>Truncates any trailing NULLs (0x00 characters) when loading data into a variable-length field. If this option is not specified, NULLs are kept.</p> <p>This option cannot be specified together with striptblanks. These are mutually exclusive options.</p> <p>Note: This option replaces the obsolete padwithzero option, which is supported for back-level compatibility only.</p>
DEL (Delimited ASCII) File Format	
chardelx	<p>x is a single character string delimiter. The default value is a double quotation mark ("). The specified character is used in place of double quotation marks to enclose a character string.^{ab}</p> <p>The single quotation mark (') can also be specified as a character string delimiter as follows:</p> <p style="padding-left: 40px;">modified by charde1''</p>
coldelx	<p>x is a single character column delimiter. The default value is a comma (.). The specified character is used in place of a comma to signal the end of a column.^{ab}</p>
datesiso	<p>Date format. Causes all date data values to be loaded in ISO format.</p>
decplusblank	<p>Plus sign character. Causes positive decimal values to be prefixed with a blank space instead of a plus sign (+). The default action is to prefix positive decimal values with a plus sign.</p>
decptx	<p>x is a single character substitute for the period as a decimal point character. The default value is a period (.). The specified character is used in place of a period as a decimal point character.^{ab}</p>
nodoubledel	<p>Suppresses recognition of double character delimiters.</p>
IXF File Format	
forcein	<p>Directs the utility to accept data despite code page mismatches, and to suppress translation between code pages.</p> <p>Fixed length target fields are checked to verify that they are large enough for the data. If nochecklengths is specified, no checking is done, and an attempt is made to load each row.</p>

Table 8 (Page 7 of 7). Valid File Type Modifications (LOAD)

Modification	Description
nochecklengths	Used with the forcein modifier. If nochecklengths is specified, fixed length target fields are <i>not</i> checked to verify that they are large enough for the data, and an attempt is made to load each row.
<p>Note:</p> <ul style="list-style-type: none"> ^a Table 6 on page 166 lists the characters that can be used as delimiter overrides. ^b The character must be specified in the code page of the source data. The character code point (instead of the character symbol), can be specified using the syntax xJJ or 0xJJ, where JJ is the hexadecimal representation of the code point. For example, to specify the # character as a column delimiter, use one of the following: <pre>... modified by coldel# modified by coldel0x23 modified by coldelX23 ...</pre> ^c Even if the DATALINK delimiter character is a valid character within the URL syntax, it will lose its special meaning within the scope of the LOAD command. 	

Remote Files

Remote file is a base file name to which DB2 appends different extensions to create files used by other commands (for example, .msg for "LOAD QUERY" on page 301).

The remote file resides on the server machine and is accessed by the DB2 instance exclusively. Therefore, it is imperative that any file name qualification given to this parameter reflects the directory structure of the server, not the client, and that the DB2 instance owner has read and write permission on this file. In addition, the user must ensure that two loads are not issued that have the same fully-qualified remote file name.

There are several ways that the remote file name can be selected and qualified when the user has just given a partially qualified name, or no name at all:

- No remote file name is given in a load operation where the user is on the same machine as the database instance. In this case, the load utility will use the name *db2utmp* and qualify it with the current working directory of the user. Two loads from the same directory with this option will clash on the use of the remote file name, therefore this option is not recommended.
- No remote file name is given in a load operation, where the user is on a different machine than the database instance. In this case, the load utility will generate a name that will reside in the database directory. This effectively prevents the user from using the load query facility, since it requires the name of the remote file. In addition, the file name generated is not guaranteed to be unique, and therefore clashes may occur between different load operations. Therefore this option is not recommended.

LOAD

- A non-fully-qualified file name is given in a load operation, where the user is on the same machine as the database instance. In this case the name is qualified by using the current directory of the user. The user must ensure that two loads are not issued from the same directory with the same remote file name.
- A non-fully-qualified file name is given in a load operation, where the user is on a different machine than the database instance. In this case the load utility will reject the file name. It must be fully qualified from the client.
- A fully-qualified file name is given in a load operation. This will be the file name used. The user must ensure that two loads are not issued with the same remote file name. This is the recommended usage.

Note: In an MPP system, the remote file must reside on a local disk, not on an NFS mount. If the file is on an NFS mount, there will be a significant performance decrement during the load operation.

See Also

“LOAD QUERY” on page 301

“QUIESCE TABLESPACES FOR TABLE” on page 328.

LOAD QUERY

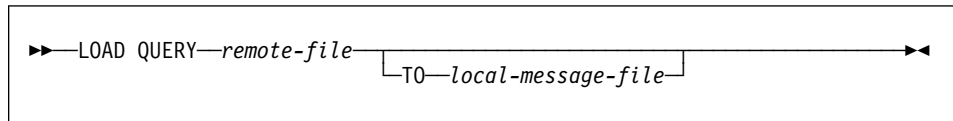
Checks the status of LOAD during processing.

Authorization

None

Required Connection

Database

Command Syntax**Command Parameters**

remote-file

Specifies the base name that was used when creating temporary files during a load.

TO *local-message-file*

Specifies the destination for warning and error messages that occur during the load. This file cannot be the *message-file* specified for LOAD. If the file already exists, all messages that the load utility has generated are appended to it.

Example

A user loading a large amount of data into the STAFF table wants to check the status of the load. Assuming that the LOAD parameter *remote-file* was set as:

```
remote file /u/remotedir/rmsg/staff
```

the following commands are issued:

```
db2 connect to <database>
```

```
db2 load query /u/remotedir/rmsg/staff to /u/mydir/staff.tempmsg
```

LOAD QUERY

The output file `/u/mydir/staff.tempmsg` might look like the following:

```
SQL3500W The utility is beginning the "LOAD" phase at time
"02-13-1997 19:40:29.645353".

SQL3519W  Begin Load Consistency Point. Input record count = "0".

SQL3520W  Load Consistency Point was successful.

SQL3109N The utility is beginning to load data from file
"/u/mydir/data/staffbig.ixf".

SQL3150N The H record in the PC/IXF file has product "DB2 01.00",
date "19970111", and time "194554".

SQL3153N The T record in the PC/IXF file has name
"data/staffbig.ixf", qualifier " ", and source " ".

SQL3519W  Begin Load Consistency Point. Input record count =
"111152".

SQL3520W  Load Consistency Point was successful.

SQL3519W  Begin Load Consistency Point. Input record count =
"222304".

SQL3520W  Load Consistency Point was successful.
```

See Also

"LOAD" on page 276.

MIGRATE DATABASE

Converts previous versions of DB2 databases to current formats. Following are the database releases that are supported in the DB2 V5.0 database migration process:

- DB2 for OS/2 Version 1.x and Version 2.x to Version 5.0
- DB2 for AIX Version 1.x and Version 2.x to Version 5.0
- DB2 for HP-UX Version 2.x to Version 5.0
- DB2 for Solaris Version 2.x to Version 5.0
- DB2 for Windows NT Version 2.x to Version 5.0
- DB2 Parallel Edition Version 1.x to Version 5.0.

Attention: The database pre-migration tool, “db2ckmig - Database Pre-migration Tool” on page 17, must be run prior to DB2 Version 5 installation (on OS/2 or the Windows operating system), or before instance migration (on UNIX based systems), because it cannot be executed on DB2 Version 5. Backup all databases prior to migration, and prior to DB2 Version 5 installation on OS/2 or the Windows operating system.

For detailed information about database migration, see one of the *Quick Beginnings* books.

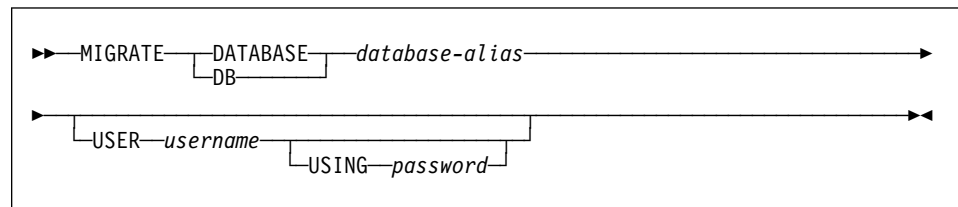
Authorization

sysadm

Required Connection

This command establishes a database connection.

Command Syntax



Command Parameters

DATABASE *database-alias*

Specifies the alias of the database to be migrated to the currently installed version of the database manager.

USER *username*

Identifies the user name under which the database is to be migrated.

USING *password*

The password used to authenticate the user name. If the password is omitted, but a user name was specified, the user is prompted to enter it.

MIGRATE DATABASE

Example

The following example migrates the database cataloged under the database alias sales:

```
db2 migrate database sales
```

Usage Notes

This command will only migrate a database to a newer version, and cannot be used to convert a migrated database to its previous version.

The database must be cataloged before migration.

PRECOMPILE PROGRAM

Processes an application program source file containing embedded SQL statements. A modified source file is produced, containing host language calls for the SQL statements and, by default, a package is created in the database.

Scope

This command can be issued from any node in `db2nodes.cfg`. It updates the database catalogs on the catalog node. Its effects are visible to all nodes.

Authorization

One of the following:

- *sysadm* or *dbadm* authority
- BINDADD privilege if a package does not exist, and one of:
 - IMPLICIT_SCHEMA authority on the database if the schema name of the package does not exist
 - CREATEIN privilege on the schema if the schema name of the package exists
- ALTERIN privilege on the schema if the package exists
- BIND privilege on the package if it exists.

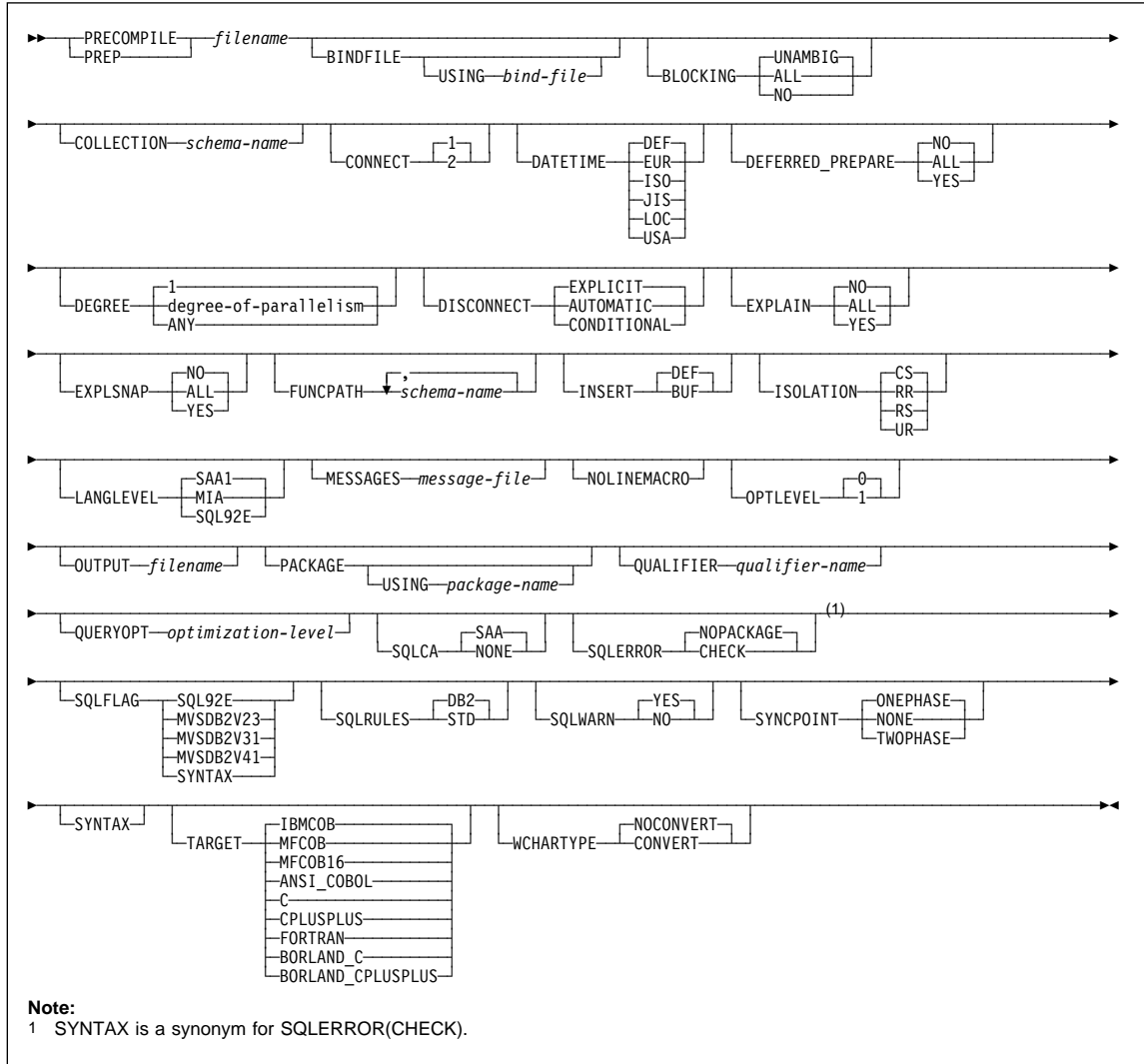
The user also needs all privileges required to compile any static SQL statements in the application. Privileges granted to groups are not used for authorization checking of static statements. If the user has *sysadm* authority, but not explicit privileges to complete the bind, the database manager grants explicit *dbadm* authority automatically.

Required Connection

Database. If implicit connect is enabled, a connection to the default database is established.

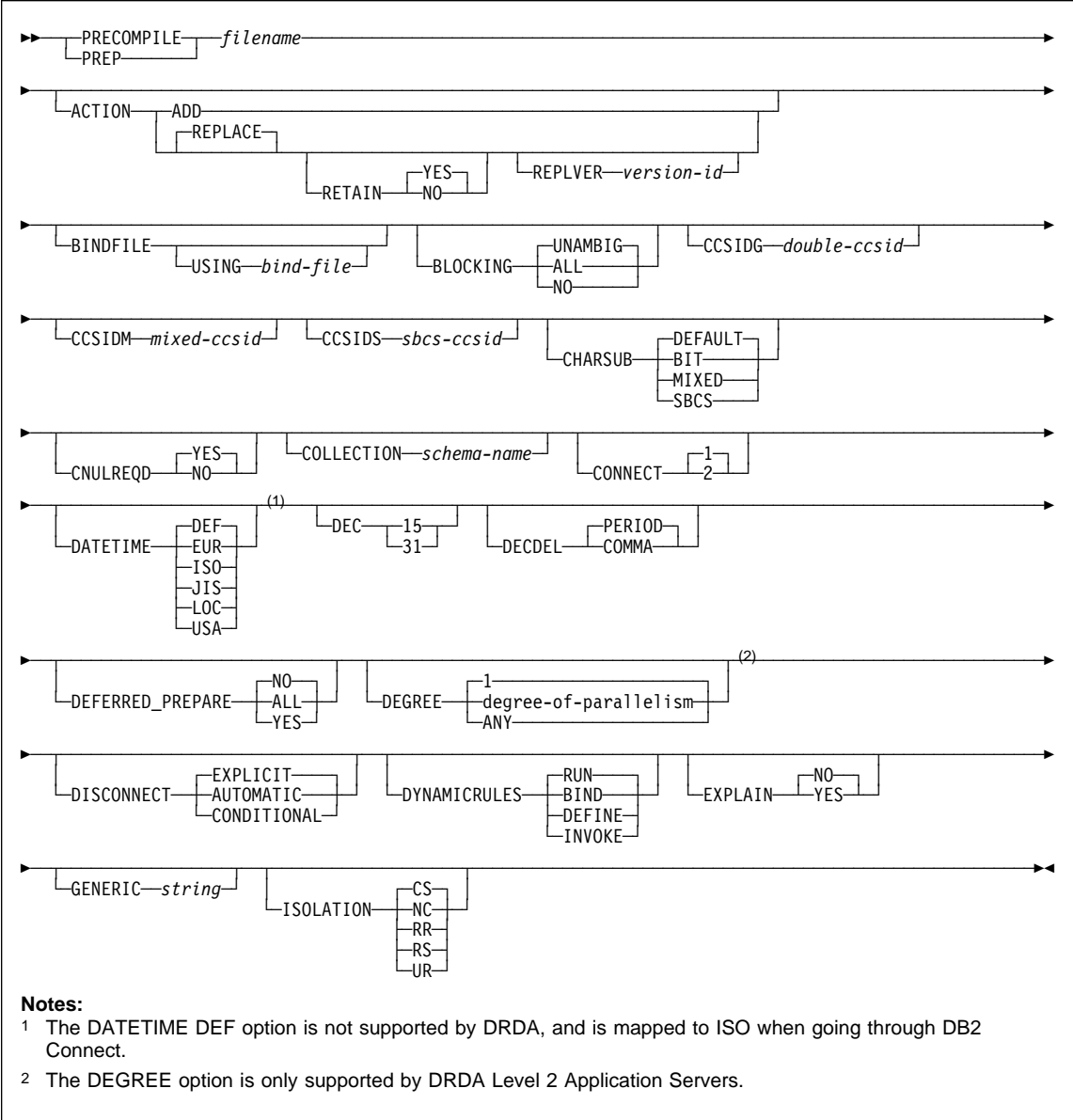
PRECOMPILE PROGRAM

Command Syntax For DB2



PRECOMPILE PROGRAM

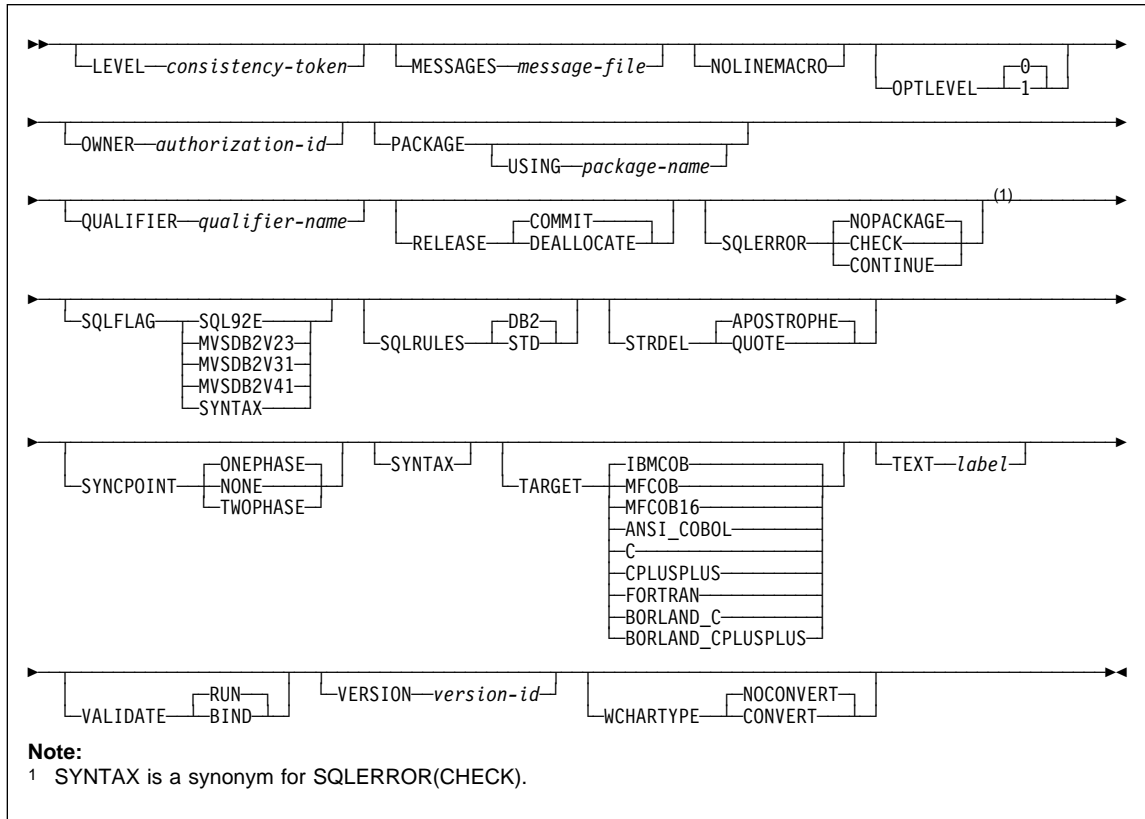
For DRDA



Continued on next page

PRECOMPILE PROGRAM

DRDA—Continued



Command Parameters

filename

Specifies the source file to be precompiled. An extension of:

- .sqc must be specified for C applications (generates a .c file)
- .sqx (OS/2 or the Windows operating system), or .sqc (UNIX based systems) must be specified for C++ applications (generates a .cxx file on OS/2 or the Windows operating system, or a .c file on UNIX based systems)
- .sqb must be specified for COBOL applications (generates a .cb1 file)
- .sqf must be specified for FORTRAN applications (generates a .for file on OS/2 or the Windows operating system, or a .f file on UNIX based systems).

The preferred extension for C++ applications containing embedded SQL on UNIX based systems is sqc; however, the sqx convention, which was invented for systems that are not case sensitive, is tolerated by UNIX based systems.

PRECOMPILE PROGRAM

ACTION

Indicates whether the package can be added or replaced. This DRDA precompile/bind option is not supported by DB2.

ADD

Indicates that the named package does not exist, and that a new package is to be created. If the package already exists, execution stops, and a diagnostic error message is returned.

REPLACE

Indicates that the old package is to be replaced by a new one with the same location, collection, and package name.

RETAIN

Indicates whether EXECUTE authorities are to be preserved when a package is replaced. If ownership of the package changes, the new owner grants the BIND and EXECUTE authority to the previous package owner.

NO

Does not preserve EXECUTE authorities when a package is replaced.

YES

Preserves EXECUTE authorities when a package is replaced.

REPLVER version-id

Replaces a specific version of a package. The version identifier specifies which version of the package is to be replaced. Maximum length is 254 characters.

BINDFILE

Results in the creation of a bind file. A package is not created unless the **package** option is also specified. If a bind file is requested, but no package is to be created, as in the following example:

```
db2 prep sample.sqc bindfile
```

object existence and authentication SQLCODEs will be treated as warnings instead of errors. This will allow a bind file to be successfully created, even if the database being used for precompilation does not have all of the objects referred to in static SQL statements within the application. The bind file can be successfully bound, creating a package, once the required objects have been created.

USING bind-file

The name of the bind file that is to be generated by the precompiler. The file name must have an extension of .bnd. If a file name is not entered, the precompiler uses the name of the program (entered as the *filename* parameter), and adds

PRECOMPILE PROGRAM

the .bnd extension. If a path is not provided, the bind file is created in the current directory.

BLOCKING

For information about row blocking, see the *Administration Guide* or the *Embedded SQL Programming Guide*.

ALL

Specifies to block for:

- Read-only cursors
- Cursors not specified as FOR UPDATE OF

Ambiguous cursors are treated as read-only.

NO

Specifies not to block any cursors. Ambiguous cursors are treated as updateable.

UNAMBIG

Specifies to block for:

- Read-only cursors
- Cursors not specified as FOR UPDATE OF

Ambiguous cursors are treated as updateable.

CCSIDG *double-ccsid*

An integer specifying the coded character set identifier (CCSID) to be used for double byte characters in character column definitions (without a specific CCSID clause) in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

CCSIDM *mixed-ccsid*

An integer specifying the coded character set identifier (CCSID) to be used for mixed byte characters in character column definitions (without a specific CCSID clause) in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

CCSIDS *sbc-ccsid*

An integer specifying the coded character set identifier (CCSID) to be used for single byte characters in character column definitions (without a specific CCSID clause) in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

CHARSUB

Designates the default character sub-type that is to be used for column definitions in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2.

BIT

Use the FOR BIT DATA SQL character sub-type in all new character columns for which an explicit sub-type is not specified.

PRECOMPILE PROGRAM

DEFAULT

Use the target system defined default in all new character columns for which an explicit sub-type is not specified.

MIXED

Use the FOR MIXED DATA SQL character sub-type in all new character columns for which an explicit sub-type is not specified.

SBCS

Use the FOR SBCS DATA SQL character sub-type in all new character columns for which an explicit sub-type is not specified.

CNULREQD

This option is related to the **langlevel** precompile option, which is not supported by DRDA. It is valid only if the bind file is created from a C or a C++ application. This DRDA bind option is not supported by DB2.

NO

The application was coded on the basis of the **langlevel** SAA1 precompile option with respect to the null terminator in C string host variables.

YES

The application was coded on the basis of the **langlevel** MIA precompile option with respect to the null terminator in C string host variables.

COLLECTION *schema-name*

Specifies an 8-character collection identifier for the package. If not specified, the authorization identifier for the user processing the package is used.

CONNECT

1

Specifies that a CONNECT statement is to be processed as a type 1 CONNECT.

2

Specifies that a CONNECT statement is to be processed as a type 2 CONNECT.

DATETIME

Specifies the date and time format to be used. For more information about date and time formats, see the *SQL Reference*.

DEF

Use a date and time format associated with the country code of the database.

EUR

Use the IBM standard for Europe date and time format.

ISO

Use the date and time format of the International Standards Organization.

PRECOMPILE PROGRAM

JIS

Use the date and time format of the Japanese Industrial Standard.

LOC

Use the date and time format in local form associated with the country code of the database.

USA

Use the IBM standard for U.S. date and time format.

DEC

Specifies the maximum precision to be used in decimal arithmetic operations. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

15

15-digit precision is used in decimal arithmetic operations.

31

31-digit precision is used in decimal arithmetic operations.

DECDEL

Designates whether a period (.) or a comma (,) will be used as the decimal point indicator in decimal and floating point literals. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

COMMA

Use a comma (,) as the decimal point indicator.

PERIOD

Use a period (.) as the decimal point indicator.

DEFERRED_PREPARE

Provides a performance enhancement when accessing DB2 common server databases or DRDA databases. This option combines the SQL PREPARE statement flow with the associated OPEN, DESCRIBE, or EXECUTE statement flow to minimize inter-process or network flow.

NO

The PREPARE statement will be executed at the time it is issued.

YES

Execution of the PREPARE statement will be deferred until the corresponding OPEN, DESCRIBE, or EXECUTE statement is issued.

The PREPARE statement will not be deferred if it uses the INTO clause, which requires an SQLDA to be returned immediately. However, if the PREPARE INTO statement is issued for a cursor that does not use any parameter markers, the processing will be optimized by pre-OPENing the cursor when the PREPARE is executed.

PRECOMPILE PROGRAM

ALL

Same as YES, except that a PREPARE INTO statement which contains parameter markers *is* deferred. If a PREPARE INTO statement does not contain parameter markers, pre-OPENing of the cursor will still be performed.

If the PREPARE statement uses the INTO clause to return an SQLDA, the application must not reference the content of this SQLDA until the OPEN, DESCRIBE, or EXECUTE statement is issued and returned.

DEGREE

Specifies the degree of parallelism for the execution of static SQL statements in an SMP system. This option does not affect CREATE INDEX parallelism.

1

The execution of the statement will not use parallelism.

degree-of-parallelism

Specifies the degree of parallelism with which the statement can be executed, a value between 2 and 32767 (inclusive).

ANY

Specifies that the execution of the statement can involve parallelism using a degree determined by the database manager.

DISCONNECT

AUTOMATIC

Specifies that all database connections are to be disconnected at commit.

CONDITIONAL

Specifies that the database connections that have been marked RELEASE or have no open WITH HOLD cursors are to be disconnected at commit.

EXPLICIT

Specifies that only database connections that have been explicitly marked for release by the RELEASE statement are to be disconnected at commit.

DYNAMICRULES

Specifies which authorization identifier to use when dynamic SQL in a package is executed. This DRDA precompile/bind option is not supported by DB2.

BIND

Indicates that the authorization identifier used for the execution of dynamic SQL is the package owner.

PRECOMPILE PROGRAM

RUN

Indicates that the authorization identifier used for the execution of dynamic SQL is the *authid* of the person executing the package.

DEFINE

Indicates that the authorization identifier used for the execution of dynamic SQL statements in a UDF or stored procedure is the definer of the UDF or stored procedure.

INVOKE

Indicates that the authorization identifier used for the execution of dynamic SQL statements in a UDF or stored procedure is the invoker of the UDF or stored procedure.

EXPLAIN

Stores information in the Explain tables about the access plans chosen for each SQL statement in the package. DRDA does not support the ALL value for this option.

NO

Explain information will not be captured.

YES

Explain tables will be populated with information about the chosen access plan.

ALL

Explain information for each eligible static SQL statement will be placed in the Explain tables. In addition, Explain information will be gathered for eligible dynamic SQL statements at run time, even if the CURRENT EXPLAIN SNAPSHOT register is set to NO. For more information about special registers, see the *SQL Reference*.

Note: This value for EXPLAIN is not supported by DRDA.

EXPLSNAP

Stores Explain Snapshot information in the Explain tables. This DB2 precompile/bind option is not supported by DRDA.

NO

An Explain Snapshot will not be captured.

YES

An Explain Snapshot for each eligible static SQL statement will be placed in the Explain tables.

ALL

An Explain Snapshot for each eligible static SQL statement will be placed in the Explain tables. In addition, Explain Snapshot information will be gathered for eligible dynamic SQL statements at run time, even if the CURRENT EXPLAIN SNAPSHOT register is set to NO. For more information about special registers, see the *SQL Reference*.

PRECOMPILE PROGRAM

FUNCPATH

Specifies the function path to be used in resolving user-defined distinct types and functions in static SQL. If this option is not specified, the default function path is "SYSIBM","SYSFUN",USER where USER is the value of the USER special register. This DB2 precompile/bind option is not supported by DRDA.

schema-name

A short SQL identifier, either ordinary or delimited, which identifies a schema that exists at the application server. No validation that the schema exists is made at precompile or at bind time. The same schema cannot appear more than once in the function path. The number of schemas that can be specified is limited by the length of the resulting function path, which cannot exceed 254 bytes. The schema SYSIBM does not need to be explicitly specified; it is implicitly assumed to be the first schema if it is not included in the function path. For more information, see the *SQL Reference*.

INSERT

Allows a program being precompiled or bound against a DB2 Universal Database Extended Enterprise Edition server to request that data inserts be buffered to increase performance.

BUF

Specifies that inserts from an application should be buffered.

DEF

Specifies that inserts from an application should not be buffered.

GENERIC *string*

Provides a means of passing new bind options to a target DRDA database. Supports the binding of new options that are defined in the target database, but that are not known to the local command. Do not use this option to pass bind options that *are* defined in "BIND" on page 98 or "PRECOMPILE PROGRAM" on page 305. This option can substantially improve dynamic SQL performance. The syntax is as follows:

```
generic "option1 value1 option2 value2 ..."
```

Each option and value must be separated by one or more blank spaces. For example, if the target DRDA database is DB2 MVS Version 5, one could use:

```
generic "keepdynamic yes"
```

to bind the new **keepdynamic** YES option, which is not defined locally on the PRECOMPILE PROGRAM or the BIND command.

The maximum length of the string is 1023 bytes. This DRDA bind option is currently only supported by DB2 MVS Version 5; it is not supported by DB2.

PRECOMPILE PROGRAM

ISOLATION

Determines how far a program bound to this package can be isolated from the effect of other executing programs. For more information about isolation levels, see the *SQL Reference*.

CS

Specifies Cursor Stability as the isolation level.

NC

No Commit. Specifies that commitment control is not to be used. This isolation level is not supported by DB2.

RR

Specifies Repeatable Read as the isolation level.

RS

Specifies Read Stability as the isolation level. Read Stability ensures that the execution of SQL statements in the package is isolated from other application processes for rows read and changed by the application.

UR

Specifies Uncommitted Read as the isolation level.

LANGLEVEL

Specifies the SQL rules that apply for both the syntax and the semantics for both static and dynamic SQL in the application. This option is not supported by DB2 Connect. For more information about this option, see the *Embedded SQL Programming Guide*.

MIA

Select the ISO/ANS SQL92 rules as follows:

- To support error SQLCODE or SQLSTATE checking, an SQLCA must be declared in the application code.
- C null-terminated strings are padded with blanks and always include a null-terminating character, even if truncation occurs.
- The FOR UPDATE clause is optional for all columns to be updated in a positioned UPDATE.
- A searched UPDATE or DELETE requires SELECT privilege on the object table of the UPDATE or DELETE statement if a column of the object table is referenced in the search condition or on the right hand side of the assignment clause.
- A column function that can be resolved using an index (for example MIN or MAX) will also check for nulls and return warning SQLSTATE 01003 if there were any nulls.
- An error is returned when a duplicate unique constraint is included in a CREATE or ALTER TABLE statement.
- An error is returned when no privilege is granted and the grantor has no privileges on the object (otherwise a warning is returned).

PRECOMPILE PROGRAM

SAA1

Select the common IBM DB2 rules as follows:

- To support error SQLCODE or SQLSTATE checking, an SQLCA must be declared in the application code.
- C null-terminated strings are not terminated with a null character if truncation occurs.
- The FOR UPDATE clause is required for all columns to be updated in a positioned UPDATE.
- A searched UPDATE or DELETE will not require SELECT privilege on the object table of the UPDATE or DELETE statement unless a fullselect in the statement references the object table.
- A column function that can be resolved using an index (for example MIN or MAX) will not check for nulls and warning SQLSTATE 01003 is not returned.
- A warning is returned and the duplicate unique constraint is ignored.
- An error is returned when no privilege is granted.

SQL92E

Defines the ISO/ANS SQL92 rules as follows:

- To support checking of SQLCODE or SQLSTATE values, variables by this name may be declared in the host variable declare section (if neither is declared, SQLCODE is assumed during precompilation).
- C null-terminated strings are padded with blanks and always include a null-terminating character, even if truncation occurs.
- The FOR UPDATE clause is optional for all columns to be updated in a positioned UPDATE.
- A searched UPDATE or DELETE requires SELECT privilege on the object table of the UPDATE or DELETE statement if a column of the object table is referenced in the search condition or on the right hand side of the assignment clause.
- A column function that can be resolved using an index (for example MIN or MAX) will also check for nulls and return warning SQLSTATE 01003 if there were any nulls.
- An error is returned when a duplicate unique constraint is included in a CREATE or ALTER TABLE statement.
- An error is returned when no privilege is granted and the grantor has no privileges on the object (otherwise a warning is returned).

LEVEL *consistency-token*

Defines the level of a module using the consistency token. The consistency token is any alphanumeric value up to 8 characters in length. The RDB package consistency token verifies that the requester's application and the relational database package are synchronized. This DRDA precompile option is not supported by DB2.

PRECOMPILE PROGRAM

Note: This option is not recommended for general use.

MESSAGES *message-file*

Specifies the destination for warning, error, and completion status messages. A message file is created whether the bind is successful or not. If a message file name is not specified, the messages are written to standard output. If the complete path to the file is not specified, the current directory is used. If the name of an existing file is specified, the contents of the file are overwritten.

NOLINEMACRO

Suppresses the generation of the #line macros in the output .c file. Useful when the file is used with development tools which require source line information such as profiles, cross-reference utilities, and debuggers.

Note: This precompile option is used for the C/C++ programming languages only.

OPTLEVEL

Indicates whether the C/C++ precompiler is to optimize initialization of internal SQLDAs when host variables are used in SQL statements. Such optimization can increase performance when a single SQL statement (such as FETCH) is used inside a tight loop.

0

Instructs the precompiler not to optimize SQLDA initialization.

1

Instructs the precompiler to optimize SQLDA initialization. This value should not be specified if the application uses:

- pointer host variables, as in the following example:

```
exec sql begin declare section;  
char (*name)[20];  
short *id;  
exec sql end declare section;
```
- C++ data members directly in SQL statements.

For more information, see the *Embedded SQL Programming Guide*.

OUTPUT *filename*

Overrides the default name of the modified source file produced by the compiler. It can include a path.

OWNER *authorization-id*

Designates an 8-character authorization identifier for the package owner. The owner must have the privileges required to execute the SQL statements in the package. The default is the primary authorization ID of the precompile/bind process if this option has not been explicitly specified. This DRDA precompile/bind option is not supported by DB2.

PRECOMPILE PROGRAM

PACKAGE

Creates a package. If neither **package**, **bindfile**, nor **syntax** is specified, a package is created in the database by default.

USING package-name

The name of the package that is to be generated by the precompiler. If a name is not entered, the name of the application program source file (minus extension and folded to uppercase) is used. Maximum length is 8 characters.

QUALIFIER *qualifier-name*

Provides an 8-character implicit qualifier for unqualified objects contained in the package. The default is the owner's authorization ID, whether or not **owner** is explicitly specified.

QUERYOPT *optimization-level*

Indicates the desired level of optimization for all static SQL statements contained in the package. The default value is 5. For the complete range of optimization levels available, see the SET CURRENT QUERY OPTIMIZATION statement in the *SQL Reference*. This DB2 precompile/bind option is not supported by DRDA.

RELEASE

Indicates whether resources are released at each COMMIT point, or when the application terminates. This DRDA precompile/bind option is not supported by DB2.

COMMIT

Release resources at each COMMIT point. Used for dynamic SQL statements.

DEALLOCATE

Release resources only when the application terminates.

SQLCA

For FORTRAN applications only. This option is ignored if it is used with other languages.

NONE

Specifies that the modified source code is not consistent with the SAA definition.

SAA

Specifies that the modified source code is consistent with the SAA definition.

SQLERROR

Indicates whether to create a package or a bind file if an error is encountered.

CHECK

Specifies that the target system performs all syntax and semantic checks on the SQL statements being bound. A package will not be created as part of this process. If, while binding, an existing package with the same name and version

PRECOMPILE PROGRAM

is encountered, the existing package is neither dropped nor replaced even if **action replace** was specified.

CONTINUE

A package or a bind file is created even when SQL errors are encountered. This option is not supported by DB2.

NOPACKAGE

A package or a bind file is not created if an error is encountered.

SQLFLAG

Identifies and reports on deviations from the SQL language syntax specified in this option.

A bind file or a package is created only if the **bindfile** or the **package** option is specified, in addition to the **sqlflag** option.

Local syntax checking is performed only if one of the following options is specified:

- **bindfile**
- **package**
- **sqlerror check**
- **syntax**

If **sqlflag** is not specified, the flagger function is not invoked, and the bind file or the package is not affected.

SQL92E SYNTAX

The SQL statements will be checked against ANSI or ISO SQL92 Entry level SQL language format and syntax with the exception of syntax rules that would require access to the database catalog. Any deviation is reported in the precompiler listing.

MVSDB2V23 SYNTAX

The SQL statements will be checked against MVS DB2 Version 2.3 SQL language syntax. Any deviation from the syntax is reported in the precompiler listing.

MVSDB2V31 SYNTAX

The SQL statements will be checked against MVS DB2 Version 3.1 SQL language syntax. Any deviation from the syntax is reported in the precompiler listing.

MVSDB2V41 SYNTAX

The SQL statements will be checked against MVS DB2 Version 4.1 SQL language syntax. Any deviation from the syntax is reported in the precompiler listing.

SQLRULES

Specifies:

- Whether type 2 CONNECTs are to be processed according to the DB2 rules or the Standard (STD) rules based on ISO/ANS SQL92.

PRECOMPILE PROGRAM

- How a user or application can specify the format of LOB answer set columns.

DB2

- Permits the SQL CONNECT statement to switch the current connection to another established (*dormant*) connection.
- The user or application can specify the format of a LOB column only during the first fetch request.

STD

- Permits the SQL CONNECT statement to establish a *new* connection only. The SQL SET CONNECTION statement must be used to switch to a dormant connection.
- The user or application can change the format of a LOB column with each fetch request.

SQLWARN

Indicates whether warnings will be returned from the compilation of dynamic SQL statements (via PREPARE or EXECUTE IMMEDIATE), or from describe processing (via PREPARE...INTO or DESCRIBE). This DB2 precompile/bind option is not supported by DRDA.

NO

Warnings will not be returned from the SQL compiler.

YES

Warnings will be returned from the SQL compiler.

Note: SQLCODE +238 is an exception. It is returned regardless of the **sqlwarn** option value.

STRDEL

Designates whether an apostrophe (') or double quotation marks (") will be used as the string delimiter within SQL statements. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

APOSTROPHE

Use an apostrophe (') as the string delimiter.

QUOTE

Use double quotation marks (") as the string delimiter.

SYNCPOINT

Specifies how commits or rollbacks are to be coordinated among multiple database connections.

NONE

Specifies that no Transaction Manager (TM) is to be used to perform a two-phase commit, and does not enforce single updater, multiple reader. A COMMIT is sent to each

PRECOMPILE PROGRAM

participating database. The application is responsible for recovery if any of the commits fail.

ONEPHASE

Specifies that no TM is to be used to perform a two-phase commit. A one-phase commit is to be used to commit the work done by each database in multiple database transactions.

TWOPHASE

Specifies that the TM is required to coordinate two-phase commits among those databases that support this protocol.

SYNTAX

Suppresses the creation of a package or a bind file during precompilation. This option can be used to check the validity of the source file without modifying or altering existing packages or bind files. **Syntax** is a synonym for **sqlerror check**.

If **syntax** is used together with the **package** option, **package** is ignored.

TARGET

Instructs the precompiler to produce modified code tailored to one of the supported compilers on the current platform.

IBMCOB

On AIX, code is generated for the IBM COBOL Set for AIX compiler. On OS/2, code is generated for the IBM VisualAge for COBOL compiler.

MFCOB

Code is generated for the Micro Focus COBOL compiler. On OS/2 this refers to the 32-bit Micro Focus COBOL compiler. This is the default if a **target** value is not specified with the COBOL precompiler on all UNIX platforms and Windows NT.

MFCOB16

Code is generated for the 16-bit Micro Focus COBOL compiler. This value is only valid on OS/2, and is the default if a **target** value is not specified with the COBOL precompiler.

ANSI_COBOL

Code compatible with the ANS X3.23-1985 standard is generated.

C

Code compatible with the C compilers supported by DB2 on the current platform is generated.

CPLUSPLUS

Code compatible with the C++ compilers supported by DB2 on the current platform is generated.

BORLAND_C

C code is generated for the Borland C/C++ compiler. This value is only valid on OS/2.

BORLAND_CPLUSPLUS

C++ code is generated for the Borland C/C++ compiler. This value is only valid on OS/2.

PRECOMPILE PROGRAM

FORTRAN

Code compatible with the FORTRAN compilers supported by DB2 on the current platform is generated.

TEXT *label*

The description of a package. Maximum length is 255 characters. The default value is blanks. This DRDA precompile/bind option is not supported by DB2.

VALIDATE

Determines when the database manager checks for authorization errors and object not found errors. The package owner authorization ID is used for validity checking. This DRDA precompile/bind option is not supported by DB2.

BIND

Validation is performed at precompile/bind time. If all objects do not exist, or all authority is not held, error messages are produced. If **sqlerror continue** is specified, a package/bind file is produced despite the error message, but the statements in error are not executable.

RUN

Validation is attempted at bind time. If all objects exist, and all authority is held, no further checking is performed at execution time.

If all objects do not exist, or all authority is not held at precompile/bind time, warning messages are produced, and the package is successfully bound, regardless of the **sqlerror continue** option setting. However, authority checking and existence checking for SQL statements that failed these checks during the precompile/bind process may be redone at execution time.

VERSION *version-id*

Defines the version identifier for a package. The version identifier is any alphanumeric value, \$, #, @, _, -, or ., up to 254 characters in length. This DRDA precompile option is not supported by DB2.

WCHARTYPE

For details and restrictions on the use and applicability of **wchartype**, see the *Embedded SQL Programming Guide*.

CONVERT

Host variables declared using the `wchar_t` base type will be treated as containing data in `wchar_t` format. Since this format is not directly compatible with the format of graphic data stored in the database (DBCS format), input data in `wchar_t` host variables is implicitly converted to DBCS format on behalf of the application, using the ANSI C function `wcstombs()`. Similarly, output DBCS data is implicitly converted to `wchar_t`

PRECOMPILE PROGRAM

format, using `mbstowcs()`, before being stored in host variables.

NOCONVERT

Host variables declared using the `wchar_t` base type will be treated as containing data in DBCS format. This is the format used within the database for graphic data; it is, however, different from the native `wchar_t` format implemented in the C language. Using **noconvert** means that graphic data will not undergo conversion between the application and the database, which can improve efficiency. The application is, however, responsible for ensuring that data in `wchar_t` format is not passed to the database manager. When this option is used, `wchar_t` host variables should not be manipulated with the C wide character string functions, and should not be initialized with wide character literals (*L-literals*).

Usage Notes

A modified source file is produced, which contains host language equivalents to the SQL statements. By default, a package is created in the database to which a connection has been established. The name of the package is the same as the file name (minus the extension and folded to uppercase), up to a maximum of 8 characters.

Following connection to a database, PREP executes under the transaction that was started. PREP then issues a COMMIT or a ROLLBACK operation to terminate the current transaction and start another one.

Creating a package with a schema name that does not already exist will result in the implicit creation of that schema. The schema owner is SYSIBM. The CREATEIN privilege on the schema is granted to PUBLIC.

During precompilation, an Explain Snapshot is not taken unless a package is created and **explsnap** has been specified. The snapshot is put into the Explain tables of the user creating the package. Similarly, Explain table information is only captured when **explain** is specified, and a package is created.

Precompiling stops if a fatal error or more than 100 errors occur. If a fatal error does occur, PREP stops precompiling, attempts to close all files, and discards the package.

See Also

“BIND” on page 98.

PRUNE HISTORY

Deletes entries from the recovery history file.

Authorization

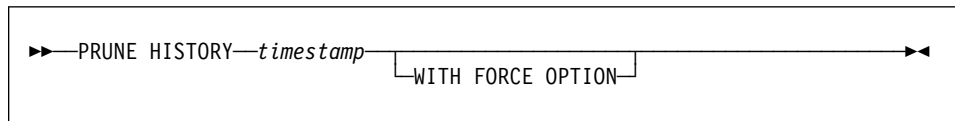
One of the following:

sysadm
sysctrl
sysmaint
dbadm

Required Connection

Database

Command Syntax



Command Parameters

timestamp

Identifies a range of entries in the recovery history file that will be deleted. A complete time stamp (in the form *yyyymmddhhnnss*), or an initial prefix (minimum *yyy*) can be specified. All entries with time stamps equal to or less than the time stamp provided are deleted from the recovery history file.

WITH FORCE OPTION

Specifies that the entries will be pruned according to the time stamp specified, even if some entries from the most recent restore set are deleted from the file.

Example

To remove the entries for all restores, loads, table space backups, and full database backups taken before and including December 1, 1994 from the recovery history file, enter:

```
db2 prune history 199412
```

QUERY CLIENT

QUERY CLIENT

Returns current connection settings for an application process.

Authorization

None

Required Connection

None

Command Syntax

```
▶—QUERY CLIENT—▶
```

Command Parameters

None

Example

The following is sample output from QUERY CLIENT:

```
The current connection settings of the application process are:

      CONNECT      = 1
      DISCONNECT   = EXPLICIT
MAX_NETBIOS_CONNECTIONS = 1
      SQLRULES     = DB2
      SYNCPOINT    = ONEPHASE
      CONNECT_NODE = CATALOG_NODE
      ATTACH_NODE  = -1
```

If `CONNECT_NODE` and `ATTACH_NODE` are not set using the `SET CLIENT` command, these parameters have values identical to that of the environment variable `DB2NODE`. If the displayed value of the `CONNECT_NODE` or the `ATTACH_NODE` parameter is `-1`, the parameter has not been set; that is, either the environment variable `DB2NODE` has not been set, or the parameter was not specified in a previously issued `SET CLIENT` command.

Usage Notes

The connection settings for an application process can be queried at any time during execution.

For information about distributed unit of work (DUOW), see the *Administration Guide*.

See Also

“SET CLIENT” on page 382.

QUIESCE TABLESPACES FOR TABLE

QUIESCE TABLESPACES FOR TABLE

Quiesces table spaces for a table. There are three valid quiesce modes: share, intent to update, and exclusive. There are three possible states resulting from the quiesce function: QUIESCED SHARE, QUIESCED UPDATE, and QUIESCED EXCLUSIVE.

Scope

In a single-node environment, this command quiesces all table spaces involved in a load operation in exclusive mode for the duration of the load. In an MPP environment, this command acts locally on a node. It quiesces only that portion of table spaces belonging to the node on which the load is performed.

Authorization

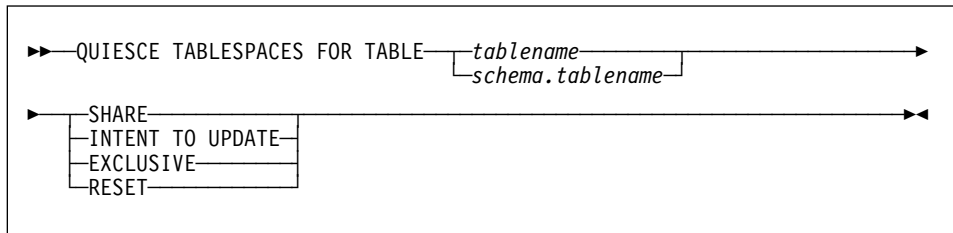
One of the following:

sysadm
sysctrl
sysmaint
dbadm

Required Connection

Database

Command Syntax



Command Parameters

TABLE

tablename

Specifies the unqualified table name. The table cannot be a system catalog table.

schema.tablename

Specifies the qualified table name. If *schema* is not provided, the authorization ID used for the database connection will be used as the *schema*. The table cannot be a system catalog table.

SHARE

Specifies that quiesce is in share mode.

QUIESCE TABLESPACES FOR TABLE

INTENT TO UPDATE

Specifies that quiesce is in intent to update mode.

EXCLUSIVE

Specifies that quiesce is in exclusive mode.

RESET

Specifies that the state of the table spaces is reset to normal.

Examples

```
db2 quiesce tablespaces for table staff share
```

```
db2 quiesce tablespaces for table boss.org intent to update
```

Usage Notes

A quiesce is a persistent lock. Its benefit is that it persists across transaction failures, connection failures, and even across system failures (such as power failure, or reboot).

A quiesce is owned by a connection. If the connection holding a table space quiesce is lost prior to the quiesce being removed, the quiesce remains on the table space, but it moves into a state of non-ownership, and is called a *phantom quiesce*. A phantom quiesce becomes "owned" by the next connection that issues the QUIESCE TABLESPACES FOR TABLE command against the same table spaces or table with the current quiesce mode. For example, if a power outage caused a load operation to be interrupted during the DELETE PHASE, the table spaces for the loaded table would be left in DELETE PENDING, QUIESCE EXCLUSIVE state. Upon database restart, the quiesces on the affected table space would be unowned (or phantom) quiesces. The user, wishing to restart the interrupted load operation, first issues a QUIESCE command for table <LOAD table name> EXCLUSIVE, establishing this connection as the owner of the quiesce. At this point, the load can be restarted.

When the quiesce share request is received, the transaction requests intent share locks for the table spaces and a share lock for the table. When the transaction obtains the locks, the state of the table spaces is changed to QUIESCED SHARE. The state is granted to the quiescer only if there is no conflicting state held by other users. The state of the table spaces is recorded in the table space table, along with the authorization ID and the database agent ID of the quiescer, so that the state is persistent.

The table cannot be changed while the table spaces for the table are in QUIESCED SHARE state. Other share mode requests to the table and table spaces will be allowed. When the transaction commits or rolls back, the locks are released, but the table spaces for the table remain in QUIESCED SHARE state until the state is explicitly reset.

When the quiesce exclusive request is made, the transaction requests super exclusive locks on the table spaces, and a super exclusive lock on the table. When the transaction obtains the locks, the state of the table spaces changes to QUIESCED EXCLUSIVE. The state of the table spaces, along with the authorization ID and the database agent ID of the quiescer, are recorded in the table space table. Since the

QUIESCE TABLESPACES FOR TABLE

table spaces are held in super exclusive mode, no other access to the table spaces is allowed. The user who invokes the quiesce function (the quiescer), however, has exclusive access to the table and the table spaces.

When a quiesce update request is made, the table spaces are locked in intent exclusive (IX) mode, and the table is locked in update (U) mode. The state of the table spaces with the quiescer is recorded in the table space table.

There is a limit of five quiescers on a table space at any given time. Since QUIESCED EXCLUSIVE is incompatible with any other state, and QUIESCED UPDATE is incompatible with another QUIESCED UPDATE, the five quiescer limit, if reached, must have at least four QUIESCED SHARE and at most one QUIESCED UPDATE.

A quiescer can upgrade the state of a table space from a less restrictive state to a more restrictive one (for example, S to U, or U to X). If a user requests a state lower than one that is already held, the original state is returned. States are not downgraded.

See Also

“LOAD” on page 276.

QUIT

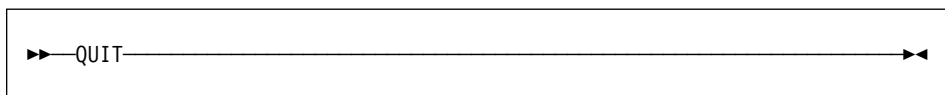
Exits the command line processor interactive input mode and returns to the operating system command prompt. If a batch file is being used to input commands to the command line processor, commands are processed until QUIT, TERMINATE, or the end-of-file is encountered.

Authorization

None

Required Connection

None

Command SyntaxA diagram showing the command syntax for QUIT. It consists of a horizontal line with a double arrowhead pointing left at the start and a double arrowhead pointing right at the end. The word "QUIT" is positioned on the line, with a vertical line segment extending from its left side to the left-pointing arrowhead.**Command Parameters**

None

Usage Notes

QUIT does not terminate the command line processor back-end process or break a database connection. CONNECT RESET breaks a connection, but does not terminate the back-end process. "TERMINATE" on page 398 does both.

REBIND

REBIND

Allows the user to recreate a package stored in the database without the need for a bind file.

Authorization

One of the following:

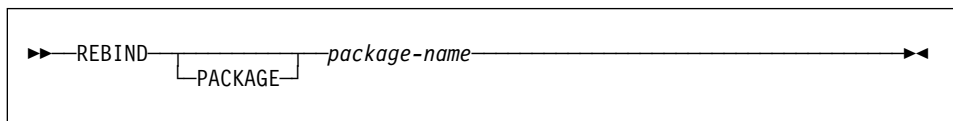
- *sysadm* or *dbadm* authority
- ALTERIN privilege on the schema
- BIND privilege on the package.

The authorization ID logged in the BOUNDBY column of the SYSCAT.PACKAGES system catalog table, which is the ID of the most recent binder of the package, is used as the binder authorization ID for the rebind, and for the default *schema* for table references in the package. Note that this default qualifier may be different from the authorization ID of the user executing the rebind request. REBIND will use the same bind options that were specified when the package was created.

Required Connection

Database. If no database connection exists, and if implicit connect is enabled, a connection to the default database is made.

Command Syntax



Command Parameters

PACKAGE *package-name*

The qualified or unqualified name that designates the package to be rebound. An unqualified package name is implicitly qualified by the current authorization ID.

Usage Notes

REBIND does not automatically commit the transaction following a successful rebind. The user must explicitly commit the transaction. This enables "what if" analysis, in which the user updates certain statistics, and then tries to rebind the package to see what changes. It also permits multiple rebinds within a unit of work.

Note: The REBIND command *will* commit the transaction if auto-commit is enabled.

This command:

- Provides a quick way to recreate a package. This enables the user to take advantage of a change in the system without a need for the original bind file. For

REBIND

example, if it is likely that a particular SQL statement can take advantage of a newly created index, the REBIND command can be used to recreate the package. REBIND can also be used to recreate packages after “RUNSTATS” on page 378 has been executed, thereby taking advantage of the new statistics.

- Provides a method to recreate inoperative packages. Inoperative packages must be explicitly rebound by invoking either the bind utility or the rebind utility. A package will be marked inoperative (the VALID column of the SYSCAT.PACKAGES system catalog will be set to X) if a function instance on which the package depends is dropped.
- Gives users control over the rebinding of invalid packages. Invalid packages will be automatically (or implicitly) rebound by the database manager when they are executed. This may result in a noticeable delay in the execution of the first SQL request for the invalid package. It may be desirable to explicitly rebind invalid packages, rather than allow the system to automatically rebind them, in order to eliminate the initial delay and to prevent unexpected SQL error messages which may be returned in case the implicit rebind fails. For example, following migration, all packages stored in the database will be invalidated by the DB2 Version 5 migration process. Given that this may involve a large number of packages, it may be desirable to explicitly rebind all of the invalid packages at one time. This explicit rebinding can be accomplished using BIND, REBIND, or the **db2rbind** tool (see “db2rbind - Rebind all Packages” on page 47).

The choice of whether to use BIND or REBIND to explicitly rebind a package depends on the circumstances. It is recommended that REBIND be used whenever the situation does not specifically require the use of BIND, since the performance of REBIND is significantly better than that of BIND. BIND *must* be used, however:

- When there have been modifications to the program (for example, when SQL statements have been added or deleted, or when the package does not match the executable for the program).
- When the user wishes to modify any of the bind options as part of the rebind. REBIND does not support any bind options. For example, if the user wishes to have privileges on the package granted as part of the bind process, BIND must be used, since it has a **grant** option.
- When the package does not currently exist in the database.
- When detection of *all* bind errors is desired. REBIND only returns the first error it detects, whereas the BIND command returns the first 100 errors that occur during binding.

REBIND is supported by DB2 Connect.

If REBIND is executed on a package that is in use by another user, the rebind will not occur until the other user's logical unit of work ends, because an exclusive lock is held on the package's record in the SYSCAT.PACKAGES system catalog table during the rebind.

When REBIND is executed, the database manager recreates the package from the SQL statements stored in the SYSCAT.STATEMENTS system catalog table.

REBIND

If REBIND encounters an error, processing stops, and an error message is returned.

REBIND will re-explain packages that were created with the **explsnap** bind option set to YES or ALL (indicated in the EXPLAIN_SNAPSHOT column in the SYSCAT.PACKAGES catalog table entry for the package) or with the **explain** bind option set to YES or ALL (indicated in the EXPLAIN_MODE column in the SYSCAT.PACKAGES catalog table entry for the package). The Explain tables used are those of the REBIND requester, not the original binder.

See Also

“BIND” on page 98

“db2rbind - Rebind all Packages” on page 47

“RUNSTATS” on page 378.

RECONCILE

Validates the references to files for the DATALINK data of a table. The rows for which the references to files cannot be established are modified in the input table.

Authorization

One of the following:

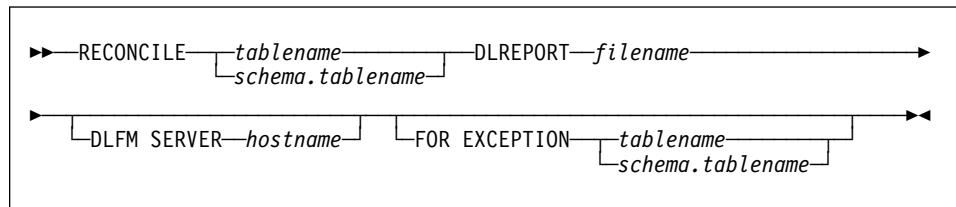
sysadm
sysctrl
sysmaint
dbadm

CONTROL privilege on the table.

Required Connection

Database

Command Syntax



Command Parameters

RECONCILE *tablename*

The table on which reconciliation is to be performed. The fully qualified name in the form: *schema.tablename*, or an alias, must be used. The *schema* is the user name under which the table was created.

DLREPORT *filename*

Specifies the file that will contain information about the files that are unlinked during reconciliation. The default name is *Reconcile_TableName_yymmdd*.

DLFM SERVER *hostname*

Specifies the DLFM server that was pre-configured for use with this database. Reconciliation will only be performed on the server specified. All other servers encountered in the table data will be ignored. Therefore, this option can be used to run RECONCILE on a particular server. The default behavior is to perform reconciliation on all servers in the table data.

FOR EXCEPTION *tablename*

This parameter is not currently used.

RECONCILE

Example

```
db2 reconcile manager dlfm server narang.almaden.ibm.com
```

Usage Notes

During reconciliation, attempts are made to link files which exist according to the table data, but which do not exist according to the DLFM metadata, if no other conflict exists.

When the reconcile utility is invoked without specifying a DLFM server name, reconciliation is performed with respect to all DATALINK data in the table. If file references cannot be established, the violating rows are not deleted from the input table, but to ensure DATALINK file reference integrity, the offending DATALINK values are NULLED. If the column is defined as not NULLable, the URL part of the DATALINK value is replaced by a zero length URL (the comment part is untouched). At the end of processing, the table is taken out of Datalink_Reconcile_Pending state.

If a DLFM server is specified, reconciliation is done only with respect to this server. In this case, other servers are not contacted, even if they are present in the DATALINK data. At the end of reconciliation, the table is taken out of Datalink_Reconcile_Pending state only if the table data has no reference to other DLFM servers. If, after performing reconciliation with respect to one or more DLFM servers for a table, the integrity of DATALINK data in the table is certain, the table can be taken out of check pending state by issuing the SET CONSTRAINTS ... IMMEDIATE UNCHECKED statement.

REDISTRIBUTE NODEGROUP

Redistributes data across the nodes in a nodegroup. The current data distribution, whether it is uniform or skewed, can be specified. The redistribution algorithm selects the partitions to be moved based on the current data distribution.

This command can only be issued from the catalog node. Use “LIST DATABASE DIRECTORY” on page 240 to determine which node is the catalog node for each database.

Scope

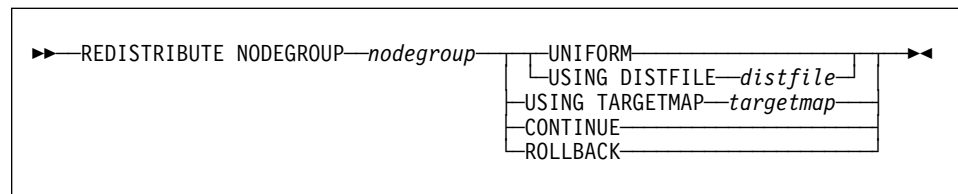
This command affects all nodes in the nodegroup.

Authorization

One of the following:

sysadm
sysctrl
dbadm

Command Syntax



Command Parameters

NODEGROUP *nodegroup*

The name of the nodegroup. This one-part name identifies a nodegroup described in the SYSNODEGROUPS catalog table. The nodegroup cannot currently be undergoing redistribution.

Note: Tables in the IBMCATGROUP and the IBMTEMPGROUP nodegroups cannot be redistributed.

UNIFORM

Specifies that the data is uniformly distributed across hash partitions (that is, every hash partition is assumed to have the same number of rows), but the same number of hash partitions do not map to each node. After redistribution, all nodes in the nodegroup have approximately the same number of hash partitions.

USING DISTFILE *distfile*

If the distribution of partitioning key values is skewed, use this option to achieve a uniform redistribution of data across the nodes of a nodegroup.

REDISTRIBUTE NODEGROUP

Use the *distfile* to indicate the current distribution of data across the 4 096 hash partitions.

Use row counts, byte volumes, or any other measure to indicate the amount of data represented by each hash partition. The utility reads the integer value associated with a partition as the weight of that partition. When a *distfile* is specified, the utility generates a target partitioning map that it uses to redistribute the data across the nodes in the nodegroup as uniformly as possible. After the redistribution, the weight of each node in the nodegroup is approximately the same (the weight of a node is the sum of the weights of all partitions that map to that node).

For example, the input distribution file may contain entries as follows:

```
10223
1345
112000
0
100
...
```

In the example, hash partition 2 has a weight of 112 000, and partition 3 (with a weight of 0) has no data mapping to it at all.

The *distfile* should contain 4 096 positive integer values in character format. The sum of the values should be less than or equal to 4 294 967 295.

If the path for *distfile* is not specified, the current directory is used.

USING TARGETMAP *targetmap*

The file specified in *targetmap* is used as the target partitioning map. Data redistribution is done according to this file. If the path is not specified, the current directory is used.

If a node included in the target map is not in the nodegroup, an error is returned. Issue ALTER NODEGROUP ADD NODE before running REDISTRIBUTE NODEGROUP.

If a node excluded from the target map *is* in the nodegroup, that node will not be included in the partitioning. Such a node can be dropped using ALTER NODEGROUP DROP NODE either before or after REDISTRIBUTE NODEGROUP.

CONTINUE

Continues a previously failed REDISTRIBUTE NODEGROUP operation. If none occurred, an error is returned.

ROLLBACK

Rolls back a previously failed REDISTRIBUTE NODEGROUP operation. If none occurred, an error is returned.

REDISTRIBUTE NODEGROUP

Usage Notes

When a redistribution operation is done, a message file is written to:

- The `$HOME/sql1lib/redis` directory on UNIX based systems, using the following format for subdirectories and file name:
database-name.nodegroup-name.timestamp.
- The `$HOME\sql1lib\redis\` directory on OS/2 or the Windows operating system, using the following format for subdirectories and file name:
database-name\first-eight-characters-of-the-nodegroup-name\date\time.

The time stamp value is the time when the command was issued.

This utility performs intermittent COMMITs during processing.

Use the ALTER NODEGROUP statement to add nodes to a nodegroup. This statement permits one to define the containers for the table spaces associated with the nodegroup. See the *SQL Reference* for details.

Note: DB2 Parallel Edition for AIX Version 1 syntax, with ADD NODE and DROP NODE options, is supported for users with *sysadm* or *sysctrl* authority. For ADD NODE, containers are created like the containers on the lowest node number of the existing nodes within the nodegroup.

All packages having a dependency on a table that has undergone redistribution are invalidated. It is recommended to explicitly rebind such packages after the redistribute nodegroup operation has completed. Explicit rebinding eliminates the initial delay in the execution of the first SQL request for the invalid package. The redistribute message file contains a list of all the tables that have undergone redistribution.

It is also recommended to update statistics by issuing "RUNSTATS" on page 378 after the redistribute nodegroup operation has completed.

Nodegroups containing replicated summary tables or tables defined with DATA CAPTURE CHANGES cannot be redistributed.

See Also

"REBIND" on page 332.

REGISTER

REGISTER

Registers the DB2 server on the network server. Adds the DB2 server's network address in a specified location on the network server.

This command is not available on the Windows operating system.

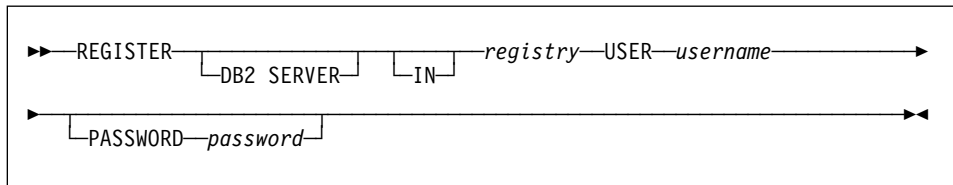
Authorization

None

Required Connection

None

Command Syntax



Command Parameters

IN *registry*

Indicates where on the network server to register the DB2 server. In this release, the only supported value is NWBINDERY (NetWare bindery).

USER *username*

User ID to log into the network server. The user ID must have SUPERVISOR or Workgroup Manager security equivalence.

PASSWORD *password*

Password used to log into the network server. The password must have SUPERVISOR or Workgroup Manager security equivalence.

Usage Notes

This command *must* be issued locally from a DB2 server. It is not supported remotely.

This command is only relevant when using file server addressing to connect a client and server.

Install DB2 and configure the database manager IPX/SPX parameters for each server instance. Then issue the REGISTER command at each DB2 server instance once, before invoking "START DATABASE MANAGER" on page 390 for the first time at each DB2 server instance. After that, if the IPX/SPX fields are reconfigured, or the server network address changes, deregister the DB2 server on the network server before making the changes, and then register it again after the changes have been made.

See Also

“DEREGISTER” on page 151.

REORGANIZE TABLE

REORGANIZE TABLE

Reorganizes a table by reconstructing the rows to eliminate fragmented data, and by compacting information.

Scope

This command affects all nodes in the nodegroup.

Authorization

One of the following:

sysadm

sysctrl

sysmaint

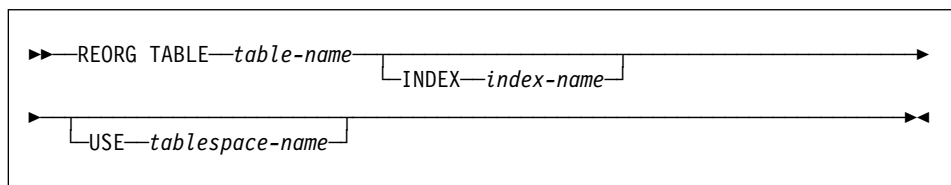
dbadm

CONTROL privilege on the table.

Required Connection

Database

Command Syntax



Command Parameters

TABLE *table-name*

Specifies the table to reorganize. The table can be in a local or a remote database. The fully qualified name or alias in the form: *schema.table-name* must be used. The *schema* is the user name under which the table was created.

INDEX *index-name*

Specifies the index to use when reorganizing the table. The fully qualified name in the form: *schema.index-name* must be used. The *schema* is the user name under which the index was created. The database manager uses the index to physically reorder the records in the table it is reorganizing. If the name of an index is not provided, the records are reorganized without regard to order.

USE *tablespace-name*

Specifies the name of a temporary table space where the database manager can temporarily store the table being reconstructed. If a table space name is not entered, the database manager stores a working copy

REORGANIZE TABLE

of the table in the table space(s) in which the table being reorganized resides.

For an 8KB table object, the page size of any temporary table space explicitly specified by the user must match the page size of the table space(s) in which the table data (including any LONG or LOB column data) resides.

Example

To reorganize the EMPLOYEE table using the temporary table space TEMPSPACE1 as a work area, enter:

```
db2 reorg table homer.employee using tempSPACE1
```

Usage Notes

Tables that have been modified so many times that data is fragmented and access performance is noticeably slow are candidates for reorganization. Use “REORGCHK” on page 345 to determine whether a table needs reorganizing. Be sure to complete all database operations and release all locks before invoking REORGANIZE TABLE. This may be done by issuing a COMMIT after closing all cursors opened WITH HOLD, or by issuing a ROLLBACK. After reorganizing a table, use “RUNSTATS” on page 378 to update the table statistics, and “REBIND” on page 332 to rebind the packages that use this table.

If the table is partitioned onto several nodes, and the table reorganization fails on any of the affected nodes, only the failing nodes will have the table reorganization rolled back.

Note: If the reorganization is not successful, temporary files should not be deleted. The database manager uses these files to recover the database.

If the name of an index is specified, the database manager reorganizes the data according to the order in the index. To maximize performance, specify an index that is often used in SQL queries. If the name of an index is *not* specified, and if a clustering index exists, the data will be ordered according to the clustering index.

The PCTFREE value of a table determines the amount of free space designated per page. If the value has not been set, the utility will fill up as much space as possible on each page.

REORGANIZE TABLE cannot be used on views.

REORGANIZE TABLE cannot be used on a DMS table while an online backup of a table space in which the table resides is being performed.

To complete a table space roll-forward recovery following a table reorganization, both data and LONG table spaces must be roll-forward enabled.

If the table contains LOB columns that do not use the COMPACT option, the LOB DATA storage object can be significantly larger following table reorganization. This can

REORGANIZE TABLE

be a result of the order in which the rows were reorganized, and the types of table spaces used (SMS/DMS).

DB2 Version 2 servers do not support down-level client requests to reorganize a table. Since pre-Version 2 servers do not support table spaces, the *tablespace-name* parameter is treated as the Version 1 *path* parameter, when Version 2 clients are used with a down-level server.

If a Version 2 client requests to reorganize a table on a Version 2 server, and that request includes a path instead of a temporary table space in the *tablespace-name* parameter (for example, an old application, specifying a temporary file path, being executed on Version 2 clients), REORG chooses a temporary table space in which to place the work files on behalf of the user. A valid temporary table space name containing a path separator character (/ or \) should not be specified, because it will be interpreted as a temporary path (pre-Version 2 request), and REORG will choose a temporary table space on behalf of the user.

See Also

“REBIND” on page 332
“REORGCHK” on page 345
“RUNSTATS” on page 378.

REORGCHK

Calculates statistics on the database to determine if tables need to be reorganized.

Scope

This command can be issued from any node in the `db2nodes.cfg` file. It can be used to update table and index statistics in the catalogs.

Authorization

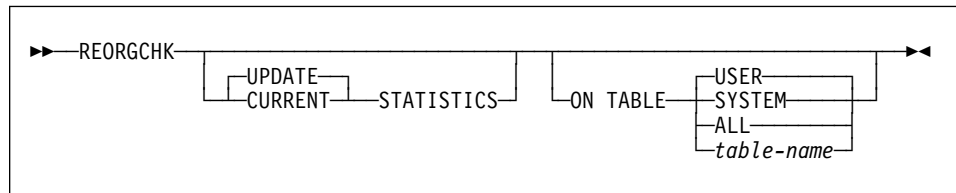
One of the following:

- `sysadm` or `dbadm` authority
- CONTROL privilege on the table.

Required Connection

Database

Command Syntax



Command Parameters

UPDATE STATISTICS

Calls the RUNSTATS routine to update table statistics, and then uses the updated statistics to determine if table reorganization is required.

If a table partition exists on the node where REORGCHK has been issued, RUNSTATS executes on this node. If a table partition does not exist on this node, the request is sent to the first node in the nodegroup that holds a partition for the table. RUNSTATS then executes on that node.

CURRENT STATISTICS

Uses the current table statistics to determine if table reorganization is required.

ON TABLE

USER

Checks the tables that are owned by the run time authorization ID.

SYSTEM

Checks the system tables.

ALL

Checks all user and system tables.

REORGCHK

table-name

Specifies the table to check. The fully qualified name or alias in the form: *schema.table-name* must be used. The *schema* is the user name under which the table was created. If the table specified is a system catalog table, the *schema* is SYSIBM.

Example

The following shows sample output from the command

```
db2 reorgchk update statistics on table system
```

run against the SAMPLE database:

```
Doing RUNSTATS ....
```

Table statistics:

```
F1: 100*OVERFLOW/CARD < 5
F2: 100*TSIZE / ((FPAGES-1) * 4020) > 70
F3: 100*NPAGES/FPAGES > 80
```

CREATOR	NAME	CARD	OV	NP	FP	TSIZE	F1	F2	F3	REORG
SYSIBM	SYSCHECKS	-	-	-	-	-	-	-	-	---
SYSIBM	SYSCOLAUTH	-	-	-	-	-	-	-	-	---
SYSIBM	SYSCOLCHECKS	-	-	-	-	-	-	-	-	---
SYSIBM	SYSCOLDIST	-	-	-	-	-	-	-	-	---
SYSIBM	SYSCOLUMNS	735	0	25	25	92610	0	95	100	---
SYSIBM	SYSCONSTDEP	-	-	-	-	-	-	-	-	---
SYSIBM	SYSDATATYPES	13	0	1	1	1027	0	-	100	---
SYSIBM	SYSDBAUTH	3	0	1	1	90	0	-	100	---
SYSIBM	SYSEVENTMONITORS	-	-	-	-	-	-	-	-	---
SYSIBM	SYSEVENTS	-	-	-	-	-	-	-	-	---
SYSIBM	SYSFUNCPARMS	254	0	6	6	21590	0	100	100	---
SYSIBM	SYSFUNCTIONS	104	0	8	8	728	0	2	100	*-
SYSIBM	SYSINDEXAUTH	2	0	1	1	112	0	-	100	---
SYSIBM	SYSINDEXES	57	17	3	5	9063	29	56	60	***
SYSIBM	SYSKEYCOLUSE	4	0	1	1	268	0	-	100	---
SYSIBM	SYSPLAN	22	0	2	2	154	0	3	100	*-
SYSIBM	SYSPLANAUTH	41	0	1	1	1804	0	-	100	---
SYSIBM	SYSPLANDEP	-	-	-	-	-	-	-	-	---
SYSIBM	SYSRELS	-	-	-	-	-	-	-	-	---
SYSIBM	SYSSECTION	4	0	1	1	260	0	-	100	---
SYSIBM	SYSSTMT	4	0	1	1	268	0	-	100	---
SYSIBM	SYSTABAUTH	68	0	2	2	3944	0	98	100	---
SYSIBM	SYSTABCONST	2	0	1	1	132	0	-	100	---
SYSIBM	SYSTABLES	69	0	6	6	483	0	2	100	*-
SYSIBM	SYSTABLESPACES	3	0	1	1	225	0	-	100	---
SYSIBM	SYSTRIGDEP	-	-	-	-	-	-	-	-	---
SYSIBM	SYSTRIGGERS	-	-	-	-	-	-	-	-	---
SYSIBM	SYSVIEWDEP	42	0	1	1	2646	0	-	100	---
SYSIBM	SYSVIEWS	32	0	5	5	3168	0	19	100	*-

REORGCHK

Index statistics:

F4: CLUSTERRATIO or normalized CLUSTERFACTOR > 80
 F5: $100 * (\text{KEYS} * (\text{ISIZE} + 10) + (\text{CARD} - \text{KEYS}) * 4) / (\text{NLEAF} * 4096) > 50$
 F6: $90 * (4000 / (\text{ISIZE} + 10) ** (\text{NLEVELS} - 2)) * 4096 / (\text{KEYS} * (\text{ISIZE} + 10) + (\text{CARD} - \text{KEYS}) * 4) < 100$

CREATOR	NAME	CARD	LEAF	LVLS	ISIZE	KEYS	F4	F5	F6	REORG

Table: SYSIBM.SYSCHECKS										
SYSIBM	IBM37	-	-	-	-	-	-	-	-	----
Table: SYSIBM.SYSCOLAUTH										
SYSIBM	IBM42	-	-	-	-	-	-	-	-	----
SYSIBM	IBM43	-	-	-	-	-	-	-	-	----
Table: SYSIBM.SYSCOLCHECKS										
SYSIBM	IBM38	-	-	-	-	-	-	-	-	----
SYSIBM	IBM39	-	-	-	-	-	-	-	-	----
Table: SYSIBM.SYSCOLDIST										
SYSIBM	IBM46	-	-	-	-	-	-	-	-	----
Table: SYSIBM.SYSCOLUMNS										
SYSIBM	IBM01	735	12	2	33	735	97	64	11	----
SYSIBM	IBM24	735	1	1	20	10	85	-	-	----
Table: SYSIBM.SYSCONSTDEP										
SYSIBM	IBM44	-	-	-	-	-	-	-	-	----
SYSIBM	IBM45	-	-	-	-	-	-	-	-	----
Table: SYSIBM.SYSDATATYPES										
SYSIBM	IBM40	13	1	1	20	13	100	-	-	----
SYSIBM	IBM41	13	1	1	2	13	100	-	-	----
Table: SYSIBM.SYSDBAUTH										
SYSIBM	IBM12	3	1	1	17	3	100	-	-	----
Table: SYSIBM.SYSEVENTMONITORS										
SYSIBM	IBM47	-	-	-	-	-	-	-	-	----
Table: SYSIBM.SYSEVENTS										
SYSIBM	IBM48	-	-	-	-	-	-	-	-	----
Table: SYSIBM.SYSFUNCPARMS										
SYSIBM	IBM31	254	2	2	30	104	100	58	77	----
SYSIBM	IBM32	254	3	2	51	154	96	79	37	----
SYSIBM	IBM33	254	1	1	6	1	100	-	-	----
Table: SYSIBM.SYSFUNCTIONS										
SYSIBM	IBM25	104	1	1	30	104	100	-	-	----
SYSIBM	IBM26	104	1	1	27	104	86	-	-	----
SYSIBM	IBM27	104	1	1	18	50	86	-	-	----
SYSIBM	IBM28	104	1	1	16	2	99	-	-	----
SYSIBM	IBM29	104	1	1	4	104	100	-	-	----
SYSIBM	IBM30	104	2	2	53	104	86	79	56	----
Table: SYSIBM.SYSINDEXAUTH										
SYSIBM	IBM17	2	1	1	47	2	100	-	-	----
SYSIBM	IBM18	2	1	1	30	2	100	-	-	----
Table: SYSIBM.SYSINDEXES										
SYSIBM	IBM02	57	1	1	17	57	100	-	-	----
SYSIBM	IBM03	57	1	1	25	57	100	-	-	----
Table: SYSIBM.SYSKEYCOLUSE										
SYSIBM	IBM35	4	1	1	57	4	100	-	-	----
SYSIBM	IBM36	4	1	1	44	2	100	-	-	----
Table: SYSIBM.SYSPLAN										
SYSIBM	IBM07	22	1	1	16	22	100	-	-	----
SYSIBM	IBM19	22	1	1	8	1	100	-	-	----
Table: SYSIBM.SYSPLANAUTH										
SYSIBM	IBM13	41	1	1	33	41	100	-	-	----
SYSIBM	IBM14	41	1	1	16	22	100	-	-	----

REORGCHK

Table: SYSIBM.SYSPLANDEP									
SYSIBM	IBM08	-	-	-	-	-	-	-	----
SYSIBM	IBM09	-	-	-	-	-	-	-	----
Table: SYSIBM.SYSRELS									
SYSIBM	IBM20	-	-	-	-	-	-	-	----
Table: SYSIBM.SYSSECTION									
SYSIBM	IBM10	4	1	1	20	4	100	-	----
Table: SYSIBM.SYSSTMT									
SYSIBM	IBM11	4	1	1	20	4	100	-	----
Table: SYSIBM.SYSTABAUTH									
SYSIBM	IBM15	68	1	1	38	68	100	-	----
SYSIBM	IBM16	68	1	1	21	68	100	-	----
Table: SYSIBM.SYSTABCONST									
SYSIBM	IBM34	2	1	1	44	2	100	-	----
Table: SYSIBM.SYSTABLES									
SYSIBM	IBM00	69	1	1	21	69	95	-	----
SYSIBM	IBM21	69	1	1	12	3	100	-	----
SYSIBM	IBM22	69	1	1	6	1	100	-	----
SYSIBM	IBM23	69	1	1	6	1	100	-	----
Table: SYSIBM.SYSTABLESPACES									
SYSIBM	IBM49	3	1	1	14	3	100	-	----
SYSIBM	IBM50	3	1	1	8	1	100	-	----
Table: SYSIBM.SYSTRIGDEP									
SYSIBM	IBM51	-	-	-	-	-	-	-	----
SYSIBM	IBM52	-	-	-	-	-	-	-	----
Table: SYSIBM.SYSTRIGGERS									
SYSIBM	IBM53	-	-	-	-	-	-	-	----
SYSIBM	IBM54	-	-	-	-	-	-	-	----
Table: SYSIBM.SYSVIEWDEP									
SYSIBM	IBM05	42	1	1	42	42	100	-	----
SYSIBM	IBM06	42	1	1	20	32	100	-	----
Table: SYSIBM.SYSVIEWS									
SYSIBM	IBM04	32	1	1	20	32	100	-	----

CLUSTERRATIO or normalized CLUSTERFACTOR (F4) will indicate REORG is necessary for indexes that are not in the same sequence as the base table. When multiple indexes are defined on a table, one or more indexes may be flagged as needing REORG. Specify the most important index for REORG sequencing.

The report headings for the table statistics (formulas 1-3) mean:

- CARD** Number of rows in base table.
- OV** (OVERFLOW) Number of overflow rows.
- NP** (NPAGES) Number of pages that contain data.
- FP** (FPAGES) Total number of pages.
- TSIZE** Table size in bytes. Calculated as the product of the number of rows in the table (CARD) and the average row length. The average row length is computed as the sum of the average column lengths (AVGCOLLEN in SYSCOLUMNS) plus 10 bytes of row overhead. For long fields and LOBs only the approximate length of the descriptor is used. The actual long field or LOB data is not counted in TSIZE.

- F1** Results of Formula 1.
- F2** Results of Formula 2.
- F3** Results of Formula 3.
- REORG** Each hyphen (-) displayed in this column indicates that the calculated results were within the set bounds of the corresponding formula, and each asterisk (*) indicates that the calculated results exceeded the set bounds of its corresponding formula.
- - or * on the left side of the column corresponds to F1 (Formula 1)
 - - or * in the middle of the column corresponds to F2 (Formula 2)
 - - or * on the right side of the column corresponds to F3 (Formula 3).
- Table reorganization is suggested when the results of the calculations exceed the bounds set by the formula.
- For example, --- indicates that, since the formula results of F1, F2, and F3 are within the set bounds of the formula, no table reorganization is suggested. The notation *-* indicates that the results of F1 and F3 suggest table reorganization, even though F2 is still within its set bounds. The notation *-- indicates that F1 is the only formula exceeding its bounds.
- The report headings for the index statistics (formulas 4-6) mean:
- CARD** Number of rows in base table.
- LEAF** Total number of index leafs (pages).
- LVLS** (LEVELS) Number of index levels.
- ISIZE** Index size, calculated from the average column length of all columns participating in the index.
- KEYS** (FULLKEYCARD) Number of unique index entries.
- F4** Results of Formula 4.
- F5** Results of Formula 5. The notation +++ indicates that the result exceeds 999, and is invalid. Rerun REORGCHK with the UPDATE STATISTICS option, or issue "RUNSTATS" on page 378, followed by the REORGCHK command.
- F6** Results of Formula 6. The notation +++ indicates that the result exceeds 999, and is invalid. Rerun REORGCHK with the UPDATE STATISTICS option, or issue "RUNSTATS" on page 378, followed by the REORGCHK command.
- REORG** Each hyphen (-) displayed in this column indicates that the calculated results were within the set bounds of the corresponding formula, and each asterisk (*) indicates that the calculated result exceeded the set bounds of its corresponding formula.
- - or * on the left side of the column corresponds to F4 (Formula 4)
 - - or * in the middle of the column corresponds to F5 (Formula 5)
 - - or * on the right side of the column corresponds to F6 (Formula 6).

REORGCHK

Table reorganization is suggested when the results of the calculations exceed the bounds set by the formula.

Usage Notes

REORGCHK calculates statistics obtained from six different formulas to determine if performance has deteriorated or can be improved by reorganizing a table.

Attention: These statistics should not be used to determine if empty tables (TSIZE=0) need reorganization. If TSIZE=0 and FPAGE>0, the table needs to be reorganized. If TSIZE=0 and FPAGE=0, no reorganization is necessary.

REORGCHK uses the following formulas to analyze the physical location of rows and the size of the table:

- Formula F1:

$$100 * \text{OVERFLOW} / \text{CARD} < 5$$

The total number of overflow rows in the table should be less than 5 percent of the total number of rows. Overflow rows can be created when rows are updated and the new rows contain more bytes than the old ones (VARCHAR fields), or when columns are added to existing tables.

- Formula F2:

$$100 * \text{TSIZE} / ((\text{FPAGES} - 1) * 4020) > 70$$

The table size in bytes (TSIZE) should be more than 70 percent of the total space allocated for the table. (There should be less than 30% free space.) Because the last page allocated is not usually filled, 1 is subtracted from FPAGES.

- Formula F3:

$$100 * \text{NPAGES} / \text{FPAGES} > 80$$

The number of pages that contain no rows at all should be less than 20 percent of the total number of pages. (Pages can become empty after rows are deleted.)

REORGCHK uses the following formulas to analyze the relationship of the indexes to the table data:

- Formula F4:

$$\text{CLUSTERRATIO or normalized CLUSTERFACTOR} > 80\%$$

The clustering ratio of an index should be greater than 80 percent. When multiple indexes are defined on one table, some of these indexes have a low cluster ratio. (The index sequence is not the same as the table sequence.) This cannot be avoided. Be sure to specify the most important index when reorganizing the table. The cluster ratio is usually not optimal for indexes that contain many duplicate keys and many entries.

- Formula F5:

$$100 * (\text{KEYS} * (\text{ISIZE} + 10) + (\text{CARD} - \text{KEYS}) * 4) / (\text{NLEAF} * 4096) > 50$$

Less than 50 percent of the space reserved for index entries should be empty (only checked when NLEAF>1).

- Formula F6:

$$90 * (4000 / (ISIZE + 10)) ** (NLEVELS - 2) * 4096 / \\ (KEYS * (ISIZE + 10) + (CARD - KEYS) * 4) < 100$$

The actual number of index entries should be more than 90% of the number of entries NLEVELS-1 can handle (only checked if NLEVELS>1).

Note: Running statistics on many tables can take time, especially if the tables are large.

See Also

“REORGANIZE TABLE” on page 342

“RUNSTATS” on page 378.

RESET ADMIN CONFIGURATION

RESET ADMIN CONFIGURATION

Resets the parameters in the database manager configuration file that are relevant to the DB2 Administration Server to the system defaults. The values are reset by node type, which is always a server with remote clients. The DB2 Administration Server is a special DB2 instance that enables remote administration of DB2 servers. The following database manager configuration parameters are reset:

- AGENT_STACK_SZ
- AUTHENTICATION
- DIAGLEVEL
- DIAGPATH
- DISCOVER
- DISCOVER_COMM
- FILESERVER
- IPX_SOCKET
- NNAME
- OBJECTNAME
- QUERY_HEAP_SZ
- SYSADM_GROUP
- SYSCTRL_GROUP
- SYSMANT_GROUP
- TPNAME
- TRUST_ALLCLNTS
- TRUST_CLNTAUTH

Note: It is not recommended that the SVCENAME parameter, set by the installation program, be modified by the user. The administration server service name is set to use the DB2 registered TCP/IP port (523).

For more information about these parameters, see “GET DATABASE MANAGER CONFIGURATION” on page 188.

Scope

This command resets the database manager configuration file, `$HOME/sqllib/db2system`. It affects all nodes that are listed in the `$HOME/sqllib/db2nodes.cfg` file.

Authorization

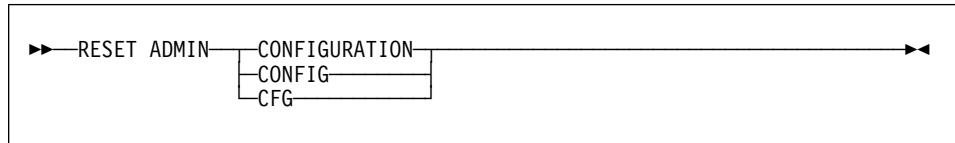
sysadm

Required Connection

None or instance. An instance attachment is not required to perform local DBM configuration operations, but is required to perform remote DBM configuration operations. To reset the database manager configuration for a remote instance, it is necessary to first attach to that instance.

RESET ADMIN CONFIGURATION

Command Syntax



Command Parameters

None

Usage Notes

To view or print a list of the admin configuration parameters, use “GET ADMIN CONFIGURATION” on page 171.

To change the value of an admin parameter, use “UPDATE ADMIN CONFIGURATION” on page 405.

For more information about these parameters, see the *Administration Guide*.

Changes to the database manager configuration file become effective only after they are loaded into memory. This occurs during execution of **db2start**.

If an error occurs, the database manager configuration file does not change.

The database manager configuration file cannot be reset if the checksum is invalid. This may occur if the database manager configuration file is changed without using the appropriate command. If this happens, the database manager must be installed again to reset the database manager configuration file.

See Also

“GET ADMIN CONFIGURATION” on page 171

“UPDATE ADMIN CONFIGURATION” on page 405.

RESET DATABASE CONFIGURATION

RESET DATABASE CONFIGURATION

Resets the configuration of a specific database to the system defaults.

Scope

This command only affects the node on which it is executed.

Authorization

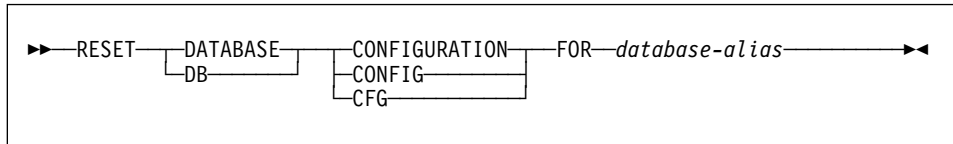
One of the following:

sysadm
sysctrl
sysmaint

Required Connection

Instance. An explicit attachment is not required. If the database is listed as remote, an instance attachment to the remote node is established for the duration of the command.

Command Syntax



Command Parameters

FOR *database-alias*

Specifies the alias of the database whose configuration is to be reset to the system defaults.

Usage Notes

To view or print a list of the database configuration parameters, use “GET DATABASE CONFIGURATION” on page 179.

To change the value of a configurable parameter, use “UPDATE DATABASE CONFIGURATION” on page 411.

For more information about these parameters, see the *Administration Guide*.

Changes to the database configuration file become effective only after they are loaded into memory. All applications must disconnect from the database before this can occur.

If an error occurs, the database configuration file does not change.

The database configuration file cannot be reset if the checksum is invalid. This may occur if the database configuration file is changed without using the appropriate

RESET DATABASE CONFIGURATION

command. If this happens, the database must be restored to reset the database configuration file.

See Also

“GET DATABASE CONFIGURATION” on page 179

“UPDATE DATABASE CONFIGURATION” on page 411.

RESET DATABASE MANAGER CONFIGURATION

RESET DATABASE MANAGER CONFIGURATION

Resets the parameters in the database manager configuration file to the system defaults. The values are reset by node type, which is always a server with remote clients.

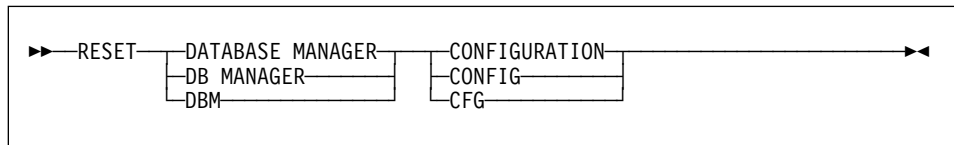
Authorization

sysadm

Required Connection

None or instance. An instance attachment is not required to perform local DBM configuration operations, but is required to perform remote DBM configuration operations. To reset the database manager configuration for a remote instance, it is necessary to first attach to that instance.

Command Syntax



Command Parameters

None

Usage Notes

To view or print a list of the database manager configuration parameters, use “GET DATABASE MANAGER CONFIGURATION” on page 188.

To change the value of a configurable parameter, use “UPDATE DATABASE MANAGER CONFIGURATION” on page 413.

For more information about these parameters, see the *Administration Guide*.

Changes to the database manager configuration file become effective only after they are loaded into memory. For a server configuration parameter, this occurs during execution of **db2start**. For a client configuration parameter, this occurs when the application is restarted. If the client is the command line processor, it is necessary to invoke “TERMINATE” on page 398.

If an error occurs, the database manager configuration file does not change.

The database manager configuration file cannot be reset if the checksum is invalid. This may occur if the database manager configuration file is changed without using the

RESET DATABASE MANAGER CONFIGURATION

appropriate command. If this happens, the database manager must be installed again to reset the database manager configuration file.

See Also

“GET DATABASE MANAGER CONFIGURATION” on page 188

“UPDATE DATABASE MANAGER CONFIGURATION” on page 413.

RESET MONITOR

RESET MONITOR

Resets the internal database system monitor data areas of a specified database, or of all active databases, to zero. The internal database system monitor data areas include the data areas for all applications connected to the database, as well as the data areas for the database itself.

Authorization

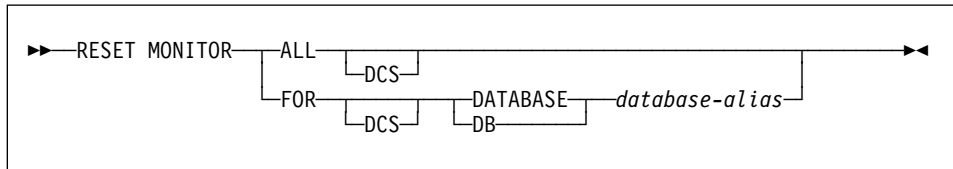
One of the following:

sysadm
sysctrl
sysmaint

Required Connection

Instance. To reset the monitor switches for a remote instance (or a different local instance), it is necessary to first attach to that instance.

Command Syntax



Command Parameters

ALL

This option indicates that the internal counters should be reset for all databases.

FOR DATABASE *database-alias*

This option indicates that only the database with alias *database-alias* should have its internal counters reset.

DCS

Depending on which clause it is specified, this keyword resets the internal counters of:

- All DCS databases
- A specific DCS database.

Usage Notes

Each process (attachment) has its own private view of the monitor data. If one user resets, or turns off a monitor switch, other users are not affected. Change the setting of the monitor switch configuration parameters to make global changes to the monitor switches (see "UPDATE DATABASE MANAGER CONFIGURATION" on page 413).

RESET MONITOR

If ALL is specified, some database manager information is also reset to maintain consistency of the returned data, and some node-level counters are reset.

To see a list of the data items that can be reset, see the *System Monitor Guide and Reference*.

See Also

“GET SNAPSHOT” on page 203

“GET MONITOR SWITCHES” on page 201.

RESTART DATABASE

RESTART DATABASE

Restarts a database that has been abnormally terminated and left in an inconsistent state. At the successful completion of RESTART DATABASE, the application remains connected to the database if the user has CONNECT privilege.

Scope

This command affects only the node on which it is executed.

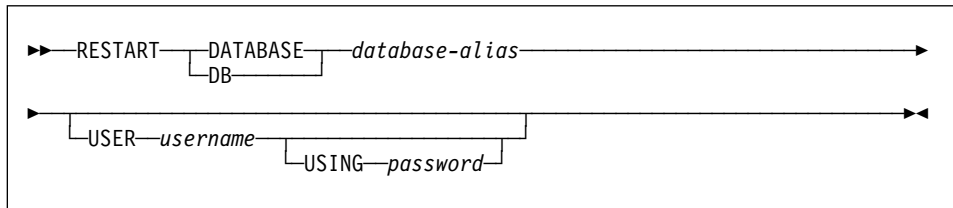
Authorization

None

Required Connection

This command establishes a database connection.

Command Syntax



Command Parameters

DATABASE *database-alias*

Identifies the database to restart.

USER *username*

Identifies the user name under which the database is to be restarted.

USING *password*

The password used to authenticate *username*. If the password is omitted, the user is prompted to enter it.

Usage Notes

Execute this command if an attempt to connect to a database returns an error message, indicating that the database must be restarted. This action occurs only if the previous session with this database terminated abnormally (due to power failure, for example).

At the completion of RESTART DATABASE, a shared connection to the database is maintained if the user has CONNECT privilege, and an SQL warning is issued if any indoubt transactions exist. In this case, the database is still usable, but if the indoubt transactions are not resolved before the last connection to the database is dropped, another RESTART DATABASE must be issued before the database can be used again. Use "LIST INDOUBT TRANSACTIONS" on page 254 to generate a list of indoubt

RESTART DATABASE

transactions. For more information about indoubt transactions, see the *Administration Guide*.

If the database is only restarted on a single node within an MPP system, a message may be returned on a subsequent database query indicating that the database needs to be restarted. This occurs because the database partition on a node on which the query depends must also be restarted. Restarting the database on all nodes solves the problem.

See Also

CONNECT TO statement in the *SQL Reference*.

RESTORE DATABASE

RESTORE DATABASE

Rebuilds a damaged or corrupted database that has been backed up using BACKUP DATABASE. The restored database is in the same state it was in when the backup copy was made. This utility can also restore to a database with a name different from the database name in the backup image (in addition to being able to restore to a new database).

The utility can also be used to restore previous versions of DB2 databases.

If, at the time of the backup operation, the database was enabled for roll-forward recovery, the database can be brought to the state it was in prior to the occurrence of the damage or corruption by issuing ROLLFORWARD DATABASE after successful execution of RESTORE DATABASE.

This utility can also restore from a table space level backup.

Scope

This command only affects the node on which it is executed.

Authorization

To restore to an existing database, one of the following:

sysadm
sysctrl
sysmaint

To restore to a new database, one of the following:

sysadm
sysctrl

Required Connection

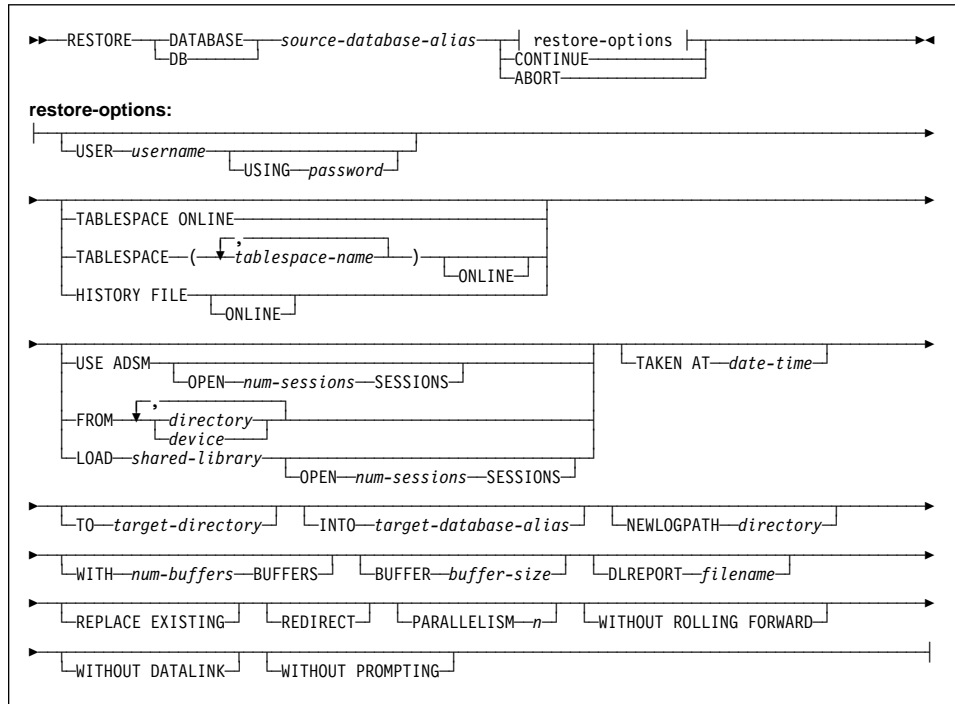
Database, to restore to an existing database.

Instance and database, to restore to a new database. The instance attachment is required to create the database.

To restore to a new remote database, it is necessary to first attach to the instance where the new database will reside.

RESTORE DATABASE

Command Syntax



Command Parameters

DATABASE *source-database-alias*

Alias of the source database from which the backup was taken.

CONTINUE

Indicates that the containers have been redefined, and that the final step in the redirected restore should be performed.

ABORT

Stops the redirected restore. Useful when an error has occurred that would require one or more steps to be repeated. After RESTORE DATABASE with the ABORT option has been issued, each step of a redirected restore must be repeated, including RESTORE DATABASE with the REDIRECT option.

USER *username*

Identifies the user name under which the database is to be restored.

USING *password*

The password used to authenticate the user name. If the password is omitted, the user is prompted to enter it.

TABLESPACE *tablespace-name*

A list of names used to specify the table spaces that are to be restored.

RESTORE DATABASE

ONLINE

This keyword, applicable only when doing a table space level restore, is specified to allow the backup to be restored online. This means that other agents can connect while the backup is being restored.

HISTORY FILE

This keyword is specified to restore the history file from the backup only.

USE ADSM

Indicates that the database is to be restored from ADSM-managed output.

OPEN *num-sessions* SESSIONS

The number of I/O sessions to be used with ADSM or the vendor product.

FROM *directory/device*

The directory or device on which the backup images reside. If USE ADSM, FROM, and LOAD are omitted, the default is the current directory.

If several items are specified, and the last item is a tape device, the user is prompted for another tape. Valid response options are:

- c** Continue. Continue using the device that generated the warning message (for example, when a new tape has been mounted)
- d** Device terminate. Stop using **only** the device that generated the warning message (for example, when there are no more tapes)
- t** Terminate. Abort the restore or backup utility.

Tape is not supported on OS/2. On OS/2, 0 or 0: can be specified to call the user exit program (see the *Administration Guide*). This option is invalid on all other platforms.

Note: Redirected restore is not allowed when a user exit program is used to perform the restore.

LOAD *shared-library*

The name of the shared library (DLL on OS/2 or the Windows operating system) containing the vendor backup and restore I/O functions to be used. It may contain the full path. If the full path is not given, it will default to the path where the user exit programs reside.

TAKEN AT *date-time*

The time stamp of the database backup. The backup image file name includes the time stamp.

TO *target-directory*

Directory of the target database. This parameter is ignored if the utility is restoring to an existing database.

INTO *target-database-alias*

Alias of the target database. If the target database does not exist, it will be created.

NEWLOGPATH *directory*

A fully qualified directory where recovery log files for the database being restored will be kept. These logs will then be used during rollforward, and any subsequent operation requiring logging. This parameter has the same function as the database configuration parameter *newlogpath*, except that its effect is limited to the RESTORE command in which it is specified.

WITH *num-buffers* BUFFERS

The number of buffers to be used.

RESTORE DATABASE

BUFFER *buffer-size*

The size, in pages, of the buffer used for the restore. The minimum value for this parameter is 16 pages; the default value is 1024 pages. If a buffer-size of 0 is specified, the value in the database manager configuration parameter *restbufsz* will be used.

The specified value is compared to the value specified during the backup. The actual restore buffer size will be an even multiple of the backup buffer size, which is equal to or greater than the backup buffer size. For example, if a backup buffer size of 1024 pages were specified, and an attempt were made to restore this backup with a buffer size of 16 pages, the actual restore buffer size would be 1024. If the specified restore buffer size were 2049, the actual restore buffer size would be 2048.

DLREPORT *filename*

The file name, if specified, must be fully qualified. The files which become unlinked during restore (as a result of a fast reconcile) will be reported.

REPLACE EXISTING

If a database with the same alias as the target database alias already exists, this parameter tells the restore utility to replace the existing database with the restored database. This is useful in scripts containing the RESTORE DATABASE command, because the CLP will not prompt the user to verify deletion of the existing database. If the WITHOUT PROMPTING parameter is specified, it is not necessary to specify REPLACE EXISTING, but in this case the command will fail if events occur that normally require user intervention.

REDIRECT

Specifies a redirected restore. To complete a redirected restore, this command should be followed by one or more SET TABLESPACE CONTAINERS commands, and then by a RESTORE DATABASE command with the CONTINUE option.

WITHOUT ROLLING FORWARD

Specifies not to place the database in roll-forward pending state after it has been successfully restored.

If, following a successful restore, the database is in roll-forward pending state, ROLLFORWARD DATABASE must be executed before the database can be used.

WITHOUT DATALINK

Specifies that any tables with DATALINK columns be placed in DataLink_Reconcile_Pending (DRP) state, and that no reconciliation of linked files is to be performed.

PARALLELISM *n*

Specifies the number of buffer manipulators to be spawned during the restore process. The default value is 1.

WITHOUT PROMPTING

Specifies that the restore will run unattended, and that any actions which normally require user intervention will instead return an error message.

RESTORE DATABASE

Examples

Following is a typical redirected restore scenario for a database whose alias is MYDB:

1. Issue a RESTORE DATABASE command with the REDIRECT option.

```
db2 restore db mydb replace existing redirect
```

After successful completion of step 1, and before completing step 3, the restore can be aborted by issuing:

```
db2 restore db mydb abort
```

2. Issue a SET TABLESPACE CONTAINERS command for each table space whose containers must be redefined. For example, on OS/2:

```
db2 set tablespace containers for 5 using  
(file 'f:\ts3con1' 20000, file 'f:\ts3con2' 20000)
```

To verify that the containers of the restored database are the ones specified in this step, issue the LIST TABLESPACE CONTAINERS command.

3. After successful completion of steps 1 and 2, issue:

```
db2 restore db mydb continue
```

This is the final step of the redirected restore.

4. If step 3 fails, or if the restore has been aborted, the redirected restore can be restarted, beginning at step 1.

Usage Notes

Database Level Restore

If restoring to an existing database, the current database configuration file is not replaced by the backup copy unless the configuration file is corrupt.

If WITHOUT ROLLING FORWARD is not specified, and the database was enabled for roll-forward recovery at the time it was backed up, the database is in roll-forward pending state after it has been successfully restored. Use “GET DATABASE CONFIGURATION” on page 179 to check the database state. If the database is in roll-forward pending state, “ROLLFORWARD DATABASE” on page 370 must be issued against the database before it can be used.

BACKUP and RESTORE can also be used to copy a database to another file system or node.

If the backup file being restored was created during an online backup, it is imperative that forward recovery be invoked at the completion of the restore. Forward recovery (using ROLLFORWARD DATABASE) will ensure that any changes which occurred during the course of the backup operation are captured to bring the database into a stable state.

For offline restore, this utility connects to the database in exclusive mode. The utility fails if any application, including the calling application, is already connected to the database that is being restored to.

RESTORE DATABASE

If an interrupt occurs during a restore, it will not be possible to successfully connect to the database until a successful restore has been performed.

The backup image must be an image that was created by the BACKUP DATABASE command, and may reside on disk, diskette (on OS/2 or the Windows operating system), tape, at the ADSM utility, or on other vendor product-managed media. Tape is not supported on OS/2.

When a database backup is restored to an existing database, the database inherits the alias and database names of the existing database. When restoring to a nonexistent database, the new database will be created with an alias and database name specified by the *target-database-alias* parameter. If a target database alias is not specified, the database will inherit the alias and database name of the backed up database.

Although a remote client may initiate a restore, the source and target always refer to entities that exist at the server.

Restoring databases may have prerequisite requirements and restrictions that are beyond the scope of this manual. For more detailed information about these conditions, see the *Administration Guide*.

Table Space Level Restore

To ensure that restored table spaces are synchronized with the rest of the database, the table spaces must be rolled forward to the end of the log (or to the point where the table spaces were last used). For this reason, table space level backup and restore can only be performed if roll-forward recovery is enabled. If roll-forward recovery is disabled at any time after a table space level backup is executed, it will not be possible to restore from the backup, and then to roll the table space forward to the current point in time. In this case, all table space level backups taken prior to that time are no longer restorable. The restore operation will fail if the user tries to restore from such a backup. In cases where it cannot be determined that the backup is invalid (if, for instance, the database has been restored and rolled forward, thus creating a new log sequence), the restore may be successful, and the broken restore set will be detected during roll-forward recovery.

Each component of a table may be backed up and restored with the table space in which it resides, independently of the other components of the table.

Table space level backup and restore cannot be run concurrently.

DB2 File Manager Considerations

When restoring to a database name or alias different from that of the backup image, tables with DATALINK columns are put in DRNP state.

For detailed information about DB2 File Manager and database recovery, see the *Administration Guide*.

RESTORE DATABASE

See Also

“BACKUP DATABASE” on page 93

“GET DATABASE CONFIGURATION” on page 179

“MIGRATE DATABASE” on page 303

“ROLLFORWARD DATABASE” on page 370.

REWIND TAPE

DB2 for Windows NT supports backup and restore to streaming tape devices. Use this command for tape rewinding.

This command is available on Windows NT only.

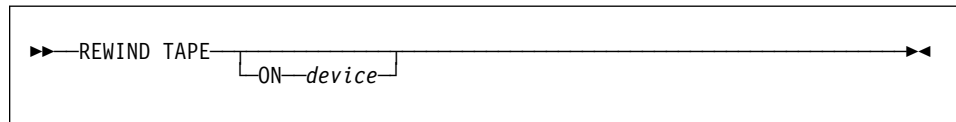
Authorization

None

Required Connection

None

Command Syntax



Command Parameters

ON *device*

Specifies a valid tape device name. The default value is `\\.\TAPE0`.

See Also

"INITIALIZE TAPE" on page 232

"SET TAPE POSITION" on page 389.

ROLLFORWARD DATABASE

ROLLFORWARD DATABASE

Recovers a database by applying transactions recorded in the database log files. Invoked after a database or a table space backup has been restored, or if any table spaces have been taken offline by the database due to a media error. The database must be recoverable (that is, either *logretain*, *userexit*, or both of these database configuration parameters must be set on) before the database can be recovered with roll-forward recovery.

Scope

In a multi-node environment, this command can only be issued from the catalog node. A database or table space rollforward command specifying a point-in-time affects all nodes that are listed in the `db2nodes.cfg` file. A database or table space rollforward command specifying end of logs affects the nodes that are specified. If no nodes are specified, it affects all nodes that are listed in the `db2nodes.cfg` file; if no roll forward is needed on a particular node, that node is ignored.

Authorization

One of the following:

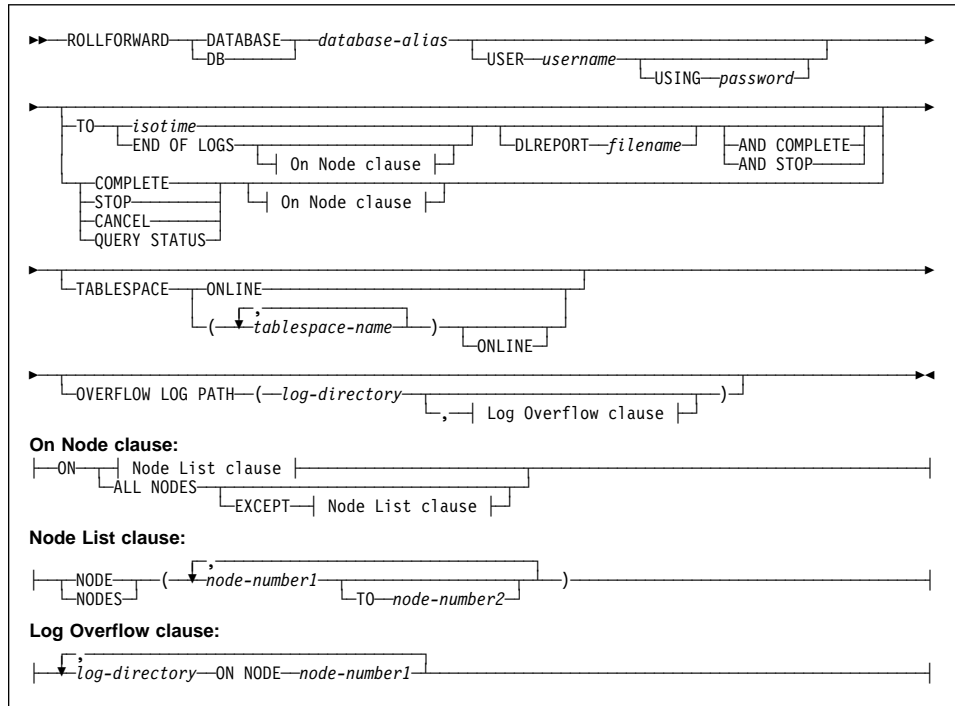
- sysadm*
- sysctrl*
- sysmaint*

Required Connection

None. This command establishes a database connection.

ROLLFORWARD DATABASE

Command Syntax



Command Parameters

DATABASE *database-alias*

The alias of the database to roll forward.

USER *username*

Identifies the user name under which the database is to be rolled forward.

USING *password*

The password used to authenticate the user name. If the password is omitted, the user is prompted to enter it.

TO

isotime

The point in time to which all committed transactions are to be rolled forward (including the transaction committed precisely at that time, as well as all transactions committed previously).

This value is specified as a time stamp, a 7-part character string that identifies a combined date and time. The format is *yyyy-mm-dd-hh.mm.ss.nnnnnn* (year, month, day, hour, minutes, seconds, microseconds), expressed in Coordinated Universal Time (CUT).

ROLLFORWARD DATABASE

The environment variable **TZ** indicates the difference between CUT and local time. For example, a **TZ** value of EST5EDT indicates that the local time zone is EST; that there is a 5-hour difference between this time zone and CUT; and that daylight savings time is observed. This observance reduces the difference from 5 to 4 hours when daylight savings time is in effect, and CUT = current time + 4.

Note: In a multi-node environment, if *isotime* is specified, roll forward recovery is performed on all nodes.

END OF LOGS

Specifies that all committed transactions from all online archive log files listed in the database configuration parameter *logpath* are to be applied.

ALL NODES

Specifies that transactions are to be rolled forward on all nodes specified in the *db2nodes.cfg* file. This is the default if a node clause is not specified.

EXCEPT

Specifies that transactions are to be rolled forward on all nodes specified in the *db2nodes.cfg* file, except those specified in the node list.

ON NODE

ON NODES

Roll forward the database on a set of nodes.

node-number1

Specifies a node number in the node list.

node-number2

Specifies the second node number, so that all nodes from *node-number1* up to and including *node-number2* are included in the node list.

DLREPORT *filename*

The file name, if specified, must be fully qualified. This file contains information about the files that become unlinked during rollforward. It is only useful if rolling forward to a quiesce point which will run fast reconcile against at least one table space.

COMPLETE

STOP

Stops the rolling forward of log records, and completes the roll-forward recovery process by rolling back any incomplete transactions and turning off the roll-forward pending state of the database. This allows access to the database or table spaces that are being rolled forward. These keywords are equivalent; specify one or the other, but not both. The keyword **AND** permits specification of multiple operations at once; for example, `db2 rollforward db sample to end of logs and complete`.

Note: When rolling table spaces forward to a point-in-time, the table spaces are placed in backup pending state.

CANCEL

Cancels the roll-forward recovery process. This leaves the database or table space(s) on all nodes on which forward recovery has been started in the restore-pending state. If the database roll-forward is not in progress

ROLLFORWARD DATABASE

(that is, the database is in rollforward-pending state), this option will change the database to restore-pending state. If a table space roll-forward is not in progress (that is, the table spaces are in rollforward-pending state), a table space list must be specified. All table spaces in the list will be changed to restore-pending state. If a table space roll-forward is in progress (that is, at least one table space is in rollforward-in-progress state), all table spaces in rollforward-in-progress state will be changed to restore-pending state. If a table space list is specified, it must include all table spaces in rollforward-in-progress state. If rolling forward to a point in time, any table space passed in will be ignored, and all table spaces that are rollforward-in-progress state will be put in restore pending state. If rolling forward to the end of logs with a table space list, only those table spaces will be put in restore-pending state.

Note: Use this option with caution.

QUERY STATUS

Lists the log files that the database manager has rolled forward, the next archive file required, and the time stamp (in CUT) of the last committed transaction since roll-forward processing began. In a multi-node environment, this status information is returned for each node. The information returned contains the following fields:

Node number

Rollforward status

Status may be database or table space rollforward pending, database or table space rollforward in progress, database or table space rollforward processing STOP, or no rollforward pending.

Next log file to be read

A string containing the name of the next required log file. In a multi-node environment, use this information if the rollforward utility fails with a return code indicating that a log file is missing or a log information mismatch has occurred.

Log files processed

A string containing the names of the processed log files that are no longer needed for recovery, and that can be removed from the directory.

Last committed transaction

A string containing a time stamp in ISO format (*yyyy-mm-dd-hh.mm.ss*). This time stamp marks the last transaction committed after the completion of roll-forward recovery. The time stamp applies to the database. For table space roll-forward, it is the time stamp of the last transaction committed to the database.

Note: QUERY STATUS is the default if the TO, STOP, COMPLETE, and CANCEL clauses are omitted.

ROLLFORWARD DATABASE

If TO, STOP, or COMPLETE are specified, this information will be displayed if the command ran successfully.

TABLESPACE

This keyword is specified for table space level roll-forward.

tablespace-name

Mandatory for table space level roll-forward to a point in time. Also allows a subset of table spaces to be specified for a roll-forward to the end of logs. In a multi-node environment, each table space in the list does not have to exist at each node that is rolling forward. If it *does* exist at the node, it must be in the correct state.

ONLINE

This keyword is specified to allow the table space level roll-forward recovery to be done online. This means that other agents are allowed to connect while roll-forward recovery is in progress.

OVERFLOW LOG PATH *log-directory*

Specifies an alternate log path to be searched for archived logs during recovery. In a multi-node environment, this is the default overflow log path for all nodes.

In a single-node environment, a relative overflow log path can be specified, but in a multi-node environment, the path must be fully qualified.

log-directory **ON NODE**

In a multi-node environment, allows a different log path to override the default overflow log path for a specific node.

Examples

Example 1

The ROLLFORWARD command permits specification of multiple operations at once, each being separated with the keyword and. For example, to roll forward to the end of logs, and complete, the separate commands:

```
db2 rollforward db sample to end of logs
db2 rollforward db sample complete
```

can be combined as follows:

```
db2 rollforward db sample to end of logs and complete
```

Example 2

Roll forward to the end of logs (two table spaces have been restored):

```
db2 rollforward db sample to end of logs
```

Note: Neither AND STOP or AND COMPLETE is needed for table space roll forward to the end of logs. Table space names are not required. If not specified, all table spaces requiring roll forward recovery will be included. If only a subset of these table spaces are to be rolled forward, their names must be specified.

ROLLFORWARD DATABASE

Example 3

After three table spaces have been restored, roll forward one to the end of logs, and the other two to a point in time, both to be done online:

```
db2 rollforward db sample to end of logs tablespace(TBS1) online
```

```
db2 rollforward db sample to 1998-04-03-14.21.56.245378 and stop  
tablespace(TBS2, TBS3) online
```

Example 4

After restoring the database, roll forward to a point in time, using OVERFLOW LOG PATH to specify the directory where the user exit saves archived logs:

```
db2 rollforward db sample to 1998-04-03-14.21.56.245378 and stop  
overflow log path (/logs)
```

Example 5 (MPP)

There are three nodes: 0, 1, and 2. Table space TBS1 is defined on all nodes, and table space TBS2 is defined on nodes 0 and 2. After restoring the database on node 1, and TBS1 on nodes 0 and 2, roll forward the database on node 1:

```
db2 rollforward db sample to end of logs and stop
```

This returns warning SQL1271 ("Database is recovered but one or more table spaces are off-line on node(s) 0 and 2.").

```
db2 rollforward db sample to end of logs
```

This rolls forward TBS1 on nodes 0 and 2. The clause TABLESPACE(TBS1) is optional in this case.

Example 6 (MPP)

After restoring table space TBS1 on nodes 0 and 2 only, roll forward TBS1 on nodes 0 and 2:

```
db2 rollforward db sample to end of logs
```

Node 1 is ignored.

```
db2 rollforward db sample to end of logs tablespace(TBS1)
```

This fails because TBS1 is not ready for roll forward on node 1. Reports SQL4906N.

```
db2 rollforward db sample to end of logs on nodes (0, 2) tablespace(TBS1)
```

This completes successfully.

```
db2 rollforward db sample to 1998-04-03-14.21.56.245378 and stop tablespace(TBS1)
```

This fails because TBS1 is not ready for roll forward on node 1; all pieces must be rolled forward together.

ROLLFORWARD DATABASE

Note: With table space roll forward to a point in time, the node clause is not accepted.

After restoring TBS1 on node 1:

```
db2 rollforward db sample to 1998-04-03-14.21.56.245378 and stop tablespace(TBS1)
```

This completes successfully.

Example 7 (MPP)

After restoring a table space on all nodes, roll forward to PIT2, but do not specify AND STOP. The ROLLFORWARD command is still in progress. Cancel and roll forward to PIT1:

```
db2 rollforward db sample to pit2 tablespace(TBS1)
db2 rollforward db sample cancel tablespace(TBS1)
```

```
** restore TBS1 on all nodes **
```

```
db2 rollforward db sample to pit1 tablespace(TBS1)
db2 rollforward db sample stop tablespace(TBS1)
```

Example 8 (MPP)

Roll forward a table space that resides on eight nodes (3 to 10) listed in the db2nodes.cfg file:

```
db2 rollforward database dwtest to end of logs tablespace (tssprodt)
```

This operation to the end of logs (not point in time) completes successfully. The nodes on which the table space resides do not have to be specified. The utility defaults to the db2nodes.cfg file.

Example 9 (MPP)

Roll forward six small table spaces that reside on a single node nodegroup (on node 6):

```
db2 rollforward database dwtest to end of logs on node (6)
tablespace(tsstore, tssbuyer, tsstime, tsswhse, tsslscat, tssvendor)
```

This operation to the end of logs (not point in time) completes successfully.

Usage Notes

The database manager uses the information stored in the archived and the active log files to reconstruct the transactions performed on the database since its last backup.

If the database is in roll-forward pending state when ROLLFORWARD DATABASE is invoked, the database will be rolled forward. Table spaces are returned to normal state after a successful database roll-forward, unless an abnormal state causes one or more table spaces to go offline.

ROLLFORWARD DATABASE

If the database is not in roll-forward pending state and no point in time is specified, any table spaces that are in rollforward-in-progress state will be rolled forward to the end of logs. If no table spaces are in rollforward-in-progress state, any table spaces that are in rollforward pending state will be rolled forward to the end of logs.

The roll-forward operation can be performed on a subset of table spaces by specifying table space names.

If rolling forward table spaces to a point in time, a subset of table spaces must be specified. Only those table spaces specified will be rolled forward. Each table space must be in roll-forward pending state or, if continuing a table space roll-forward that is already in progress, in rollforward-in-progress state.

If enabling an existing database for roll-forward recovery, change the number of primary log files to the sum of the number of primary log files and secondary log files +1. More information will be logged for LONG VARCHAR fields and LOB data in a database enabled for roll-forward recovery.

Rolling databases forward may require a load recovery using tape devices. If prompted for another tape, the user can respond with one of the following:

- c** Continue. Continue using the device that generated the warning message (for example, when a new tape has been mounted)
- d** Device terminate. Stop using the device that generated the warning message (for example, when there are no more tapes)
- t** Terminate. Terminate all devices.

Rolling databases forward may involve prerequisites and restrictions that are beyond the scope of this manual. For more detailed information, see the *Administration Guide*.

See Also

“LIST HISTORY” on page 251

“LOAD” on page 276

“RESTORE DATABASE” on page 362.

RUNSTATS

RUNSTATS

Updates statistics about the physical characteristics of a table and the associated indexes. These characteristics include number of records, number of pages, and average record length. The optimizer uses these statistics when determining access paths to the data.

This utility should be called when a table has had many updates, or after reorganizing a table.

Scope

This command can be issued from any node in the `db2nodes.cfg` file. It can be used to update the catalogs on the catalog node.

The command collects statistics for a table on the node from which it is invoked. If the table does not exist on that node, the first node in the nodegroup is selected.

Authorization

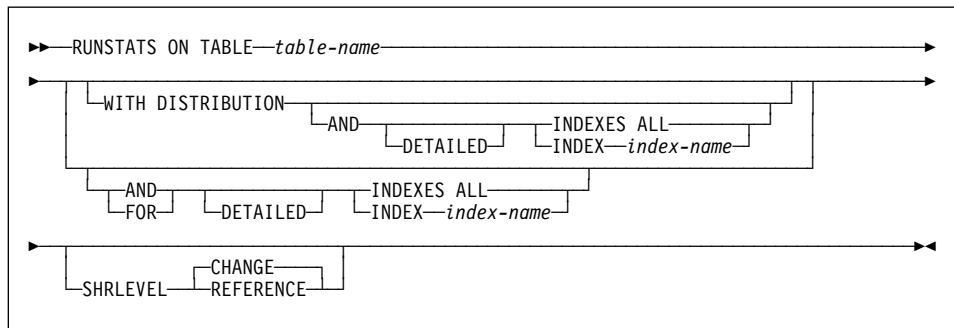
One of the following:

- `sysadm`
- `sysctrl`
- `sysmaint`
- `dbadm`
- CONTROL privilege on the table.

Required Connection

Database

Command Syntax



Command Parameters

TABLE *table-name*

The table on which to update statistics. The fully qualified name or alias in the form: `schema.table-name` must be used. The `schema` is the user name

RUNSTATS

under which the table was created. If no options are specified, only table statistics will be updated. Other users will have access to the table while the statistics are being gathered.

For row types, *table-name* must be the name of the hierarchy's root table.

WITH DISTRIBUTION

Specifies that distribution statistics are requested. The number of most frequent values collected is defined by the *num_freqvalues* database configuration parameter. The number of quantiles collected is defined by the *num_quantiles* database configuration parameter. For information about nonuniform distribution statistics, see the *Administration Guide*.

AND INDEXES ALL

Update statistics on both the table and its indexes.

AND INDEX *index-name*

Update statistics on both the table and the specified index, where *index-name* is a fully qualified name in the form: *schema.index-name*.

FOR INDEXES ALL

Update statistics on the indexes only. If statistics on the table have never been generated, the database manager calculates statistics on the table as well as on the indexes.

FOR INDEX *index-name*

Update statistics on the specified index only. If table statistics have never been generated, the database manager calculates statistics on the table as well as on the index. The index-name is a fully qualified name in the form: *schema.index-name*.

DETAILED

Calculate extended index statistics.

SHRLEVEL

CHANGE

Specifies that other users can read from and write to the table while statistics are calculated.

REFERENCE

Specifies that other users can have read-only access to the table while statistics are calculated.

Examples

Collect statistics on table only, without distribution statistics:

```
db2 runstats on table smith.table1
```

Collect statistics on table only, with distribution statistics:

```
db2 runstats on table smith.table1 with distribution
```

Collect basic statistics on indexes only:

```
db2 runstats on table smith.table1 for indexes all
```

Collect statistics on table and all indexes (basic level):

```
db2 runstats on table smith.table1 and indexes all
```

RUNSTATS

Collect statistics on table, with distribution statistics and index statistics:

```
db2 runstats on table smith.table1 with distribution and indexes all
```

Collect all possible statistics (distribution and extended index):

```
db2 runstats on table smith.table1 with distribution and detailed index
```

Collect distribution statistics on index INDEX1 only:

```
db2 runstats on table smith.table1 with distribution for index smith.index1
```

Usage Notes

Use RUNSTATS to update statistics:

- On tables that have been modified many times (for example, if a large number of updates have been made, or if a significant amount of data has been inserted or deleted)
- On tables that have been reorganized
- When a new index has been created.

After statistics have been updated, new access paths to the table can be created by rebinding the packages using “BIND” on page 98.

If index statistics are requested, and statistics have never been run on the table containing the index, statistics on both the table and indexes are calculated.

After issuing this command, a COMMIT should be issued to release the locks.

To allow new access plans to be generated, the packages that reference the target table must be rebound after issuing this command.

Statistics are collected based on the table data that is located on the database partition where the command executes. Global table statistics for an entire partitioned table are derived by multiplying the values obtained at a database partition by the number of database partitions in the nodegroup over which the table is partitioned. The global statistics are stored in the catalog tables.

The database partition from which the command is issued does not have to contain a partition for the table:

- If the command is issued from a database partition that contains a partition for the table, the utility executes at this database partition.
- If the command is issued from a database partition that does not contain a table partition, the request is sent to the first database partition in the nodegroup that holds a partition for the table. The utility then executes at this database partition.

If inconsistencies are found when running a portion of this command (resulting from activity on the table since the command was last issued), a warning message is returned. For example, if table distribution statistics were gathered on the first issue, and only index statistics are gathered on the second issue, then if inconsistencies are

RUNSTATS

detected as a result of activity on the table, the table distribution statistics are dropped. At this point, it is recommended to issue the command again to refresh the table distribution statistics.

See Also

“GET DATABASE CONFIGURATION” on page 179

“REORGANIZE TABLE” on page 342

“REORGCHK” on page 345.

SET CLIENT

SET CLIENT

Specifies connection settings for the back-end process.

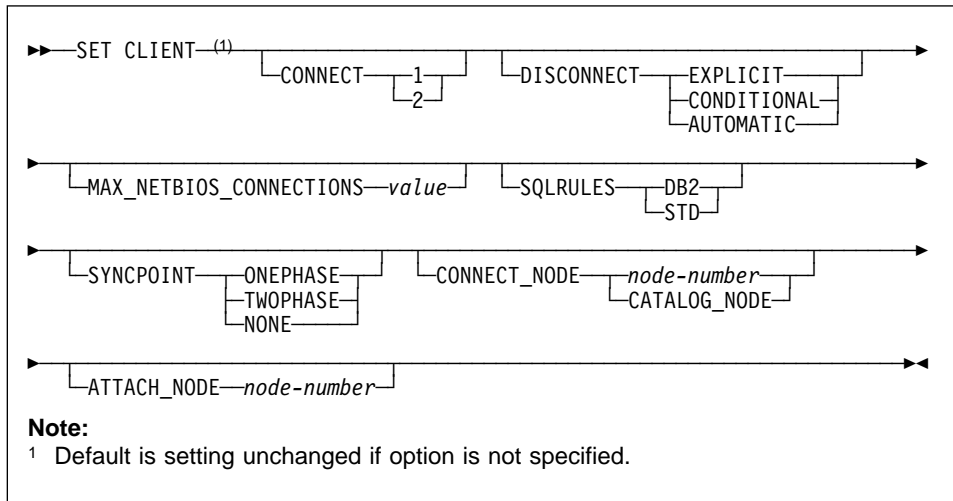
Authorization

None

Required Connection

None

Command Syntax



Command Parameters

CONNECT

1

Specifies that a CONNECT statement is to be processed as a type 1 CONNECT.

2

Specifies that a CONNECT statement is to be processed as a type 2 CONNECT.

DISCONNECT

EXPLICIT

Specifies that only database connections that have been explicitly marked for release by the RELEASE statement are to be disconnected at commit.

CONDITIONAL

Specifies that the database connections that have been marked **RELEASE** or have no open **WITH HOLD** cursors are to be disconnected at commit.

AUTOMATIC

Specifies that all database connections are to be disconnected at commit.

MAX_NETBIOS_CONNECTIONS *value*

Specifies the maximum number of concurrent connections that can be made in an application using a NetBIOS adapter. Maximum value is 254. This parameter must be set before the first NetBIOS connection is made. Changes subsequent to the first connection are ignored.

SQLRULES

DB2

Specifies that a type 2 **CONNECT** is to be processed according to the **DB2** rules.

STD

Specifies that a type 2 **CONNECT** is to be processed according to the Standard (**STD**) rules based on **ISO/ANS SQL92**.

SYNCPOINT

Specifies how commits or rollbacks are to be coordinated among multiple database connections.

ONEPHASE

Specifies that no Transaction Manager (**TM**) is to be used to perform a two-phase commit. A one-phase commit is to be used to commit the work done by each database in multiple database transactions.

TWOPHASE

Specifies that the **TM** is required to coordinate two-phase commits among those databases that support this protocol.

NONE

Specifies that no **TM** is to be used to perform a two-phase commit, and does not enforce single updater, multiple reader. A **COMMIT** is sent to each participating database. The application is responsible for recovery if any of the commits fail.

CONNECT_NODE (MPP only)

node-number

Specifies the node to which a connect is to be made. A value between zero and 999, inclusive. Overrides the value of the environment variable **DB2NODE**.

CATALOG_NODE

Specifying this value permits the client to connect to the catalog node of the database without knowing the identity of that node in advance.

SET CLIENT

ATTACH_NODE (MPP only) *node-number*

Specifies the node to which an attach is to be made. A value between zero and 999, inclusive. Overrides the value of the environment variable **DB2NODE**.

For example, if nodes 1, 2, and 3 are defined, the client only needs to be able to access one of these nodes. If only node 1 containing databases has been cataloged, and this parameter is set to 3, then the next attach attempt will result in an attachment at node 3, after an initial attachment at node 1.

Examples

To set specific values:

```
db2 set client connect 2 disconnect automatic sqlrules std
syncpoint twophase
```

To change SQLRULES back to DB2, but keep the other settings:

```
db2 set client sqlrules db2
```

Note: The connection settings revert to default values after “TERMINATE” on page 398 is issued.

Usage Notes

SET CLIENT cannot be issued if one or more connections are active.

If SET CLIENT is successful, the connections in the subsequent units of work will use the connection settings specified. If SET CLIENT is unsuccessful, the connection settings of the back-end process are unchanged.

For more information about distributed unit of work (DUOW), see the *Administration Guide*.

See Also

“QUERY CLIENT” on page 326.

SET RUNTIME DEGREE

Sets the maximum run time degree of intra-partition parallelism for SQL statements for specified active applications.

Scope

This command affects all nodes that are listed in the \$HOME/sqllib/db2nodes.cfg file.

Authorization

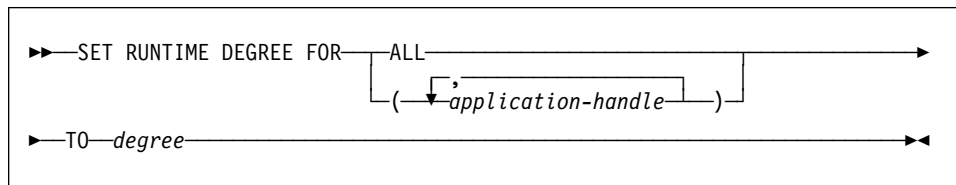
One of the following:

sysadm
sysctrl

Required Connection

Instance. To change the maximum run time degree of intra-partition parallelism on a remote server, it is first necessary to attach to that server. If no attachment exists, the SET RUNTIME DEGREE command fails.

Command Syntax



Command Parameters

FOR

ALL

The specified degree will apply to all applications.

application-handle

Specifies the agent to which the new degree applies. List the values using "LIST APPLICATIONS" on page 236.

TO *degree*

The maximum run time degree of intra-partition parallelism.

Example

The following example sets the maximum run time degree of parallelism for two users, with *application-handle* values of 41408 and 55458, to 4:

```
db2 SET RUNTIME DEGREE FOR ( 41408, 55458 ) TO 4
```

SET RUNTIME DEGREE

Usage Notes

This command provides a mechanism to modify the maximum degree of parallelism for active applications. It can be used to override the value that was determined at SQL statement compilation time.

The run time degree of intra-partition parallelism specifies the maximum number of parallel operations that will be used when the statement is executed. The degree of intra-partition parallelism for an SQL statement can be specified at statement compilation time using the CURRENT DEGREE special register or the **degree** bind option. The maximum run time degree of intra-partition parallelism for an active application can be specified using the SET RUNTIME DEGREE command. The *max_querydegree* database manager configuration parameter specifies the maximum run time degree for any SQL statement executing on this instance of the database manager.

The actual run time degree will be the lowest of:

- the *max_querydegree* configuration parameter
- the application run time degree
- the SQL statement compilation degree.

SET TABLESPACE CONTAINERS

A *redirected restore* is a restore in which the set of table space containers for the restored database is different from the set of containers for the original database at the time the backup was done. This command permits the addition, change, or removal of table space containers for a database that is to be restored. If, for example, one or more containers become inaccessible for any reason, the restore fails if it is not redirected to different containers.

Note: A redirected restore is not allowed when a user exit program is used to perform the restore.

Authorization

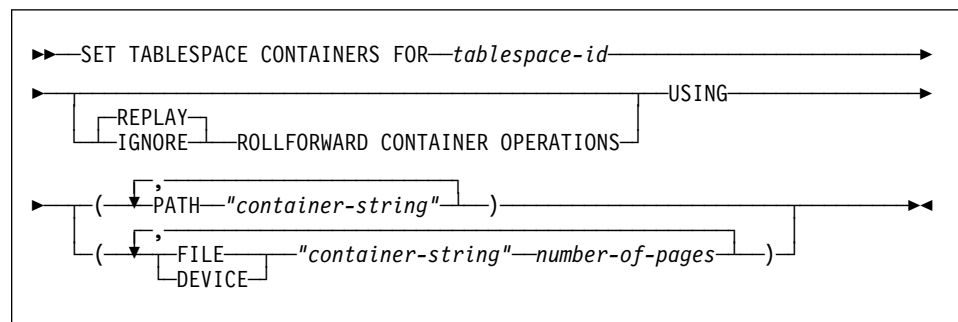
One of the following:

sysadm
sysctrl

Required Connection

Database

Command Syntax



Command Parameters

FOR *tablespace-id*

An integer that uniquely represents a table space used by the database being restored.

REPLAY ROLLFORWARD CONTAINER OPERATIONS

Specifies that any ALTER TABLESPACE operation issued against this table space since the database was backed up is to be redone during a subsequent roll forward of the database.

IGNORE ROLLFORWARD CONTAINER OPERATIONS

Specifies that ALTER TABLESPACE operations in the log are to be ignored when performing a roll forward.

SET TABLESPACE CONTAINERS

USING PATH *"container-string"*

For an SMS table space, identifies one or more containers that will belong to the table space and into which the table space data will be stored. It is an absolute or relative directory name. If the directory name is not absolute, it is relative to the database directory. The string cannot exceed 240 bytes in length.

USING FILE/DEVICE *"container-string" number-of-pages*

For a DMS table space, identifies one or more containers that will belong to the table space and into which the table space data will be stored. The container type (either FILE or DEVICE) and its size (in 4KB pages) are specified. A mixture of file and device containers can be specified. The string cannot exceed 254 bytes in length.

For a file container, the string must be an absolute or relative file name. If the file name is not absolute, it is relative to the database directory.

For a device container, the string must be a device name. The device must already exist. Device containers are not supported on OS/2.

Example

See the example in "RESTORE DATABASE" on page 362.

Usage Notes

This command is used in conjunction with "RESTORE DATABASE" on page 362.

A backup of a database, or one or more table spaces, keeps a record of all the table space containers in use by the table spaces being backed up. During a restore, all containers listed in the backup are checked to see if they currently exist and are accessible. If one or more of the containers is inaccessible for any reason, the restore will fail. In order to allow a restore in such a case, the redirecting of table space containers is supported during the restore. This support includes adding, changing, or removing of table space containers. It is this command that allows the user to add, change or remove those containers. For more information, see the *Administration Guide*.

See Also

"BACKUP DATABASE" on page 93

"RESTORE DATABASE" on page 362

"ROLLFORWARD DATABASE" on page 370.

SET TAPE POSITION

DB2 for Windows NT supports backup and restore to streaming tape devices. Use this command for tape positioning.

This command is available on Windows NT only.

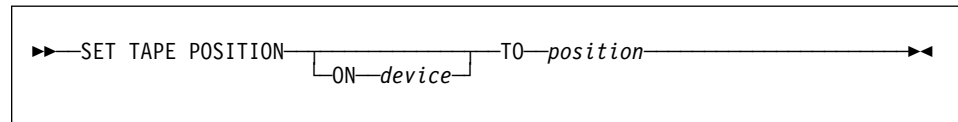
Authorization

None

Required Connection

None

Command Syntax



Command Parameters

ON *device*

Specifies a valid tape device name. The default value is `\\.\TAPE0`.

TO *position*

Specifies the mark at which the tape is to be positioned. DB2 for Windows NT writes a tape mark after every backup. A value of 1 specifies the first position, 2 specifies the second position, and so on. If the tape is positioned at tape mark 1, archive 2 is positioned to be restored.

See Also

"INITIALIZE TAPE" on page 232

"REWIND TAPE" on page 369.

START DATABASE MANAGER

START DATABASE MANAGER

Starts the current database manager instance background processes on a single node or on all the nodes defined in a multi-node environment.

This command is not valid on a client.

Scope

In a multi-node environment, this command affects all nodes that are listed in the `$HOME/sqllib/db2nodes.cfg` file, unless the `nodenum` parameter is used.

Authorization

One of the following:

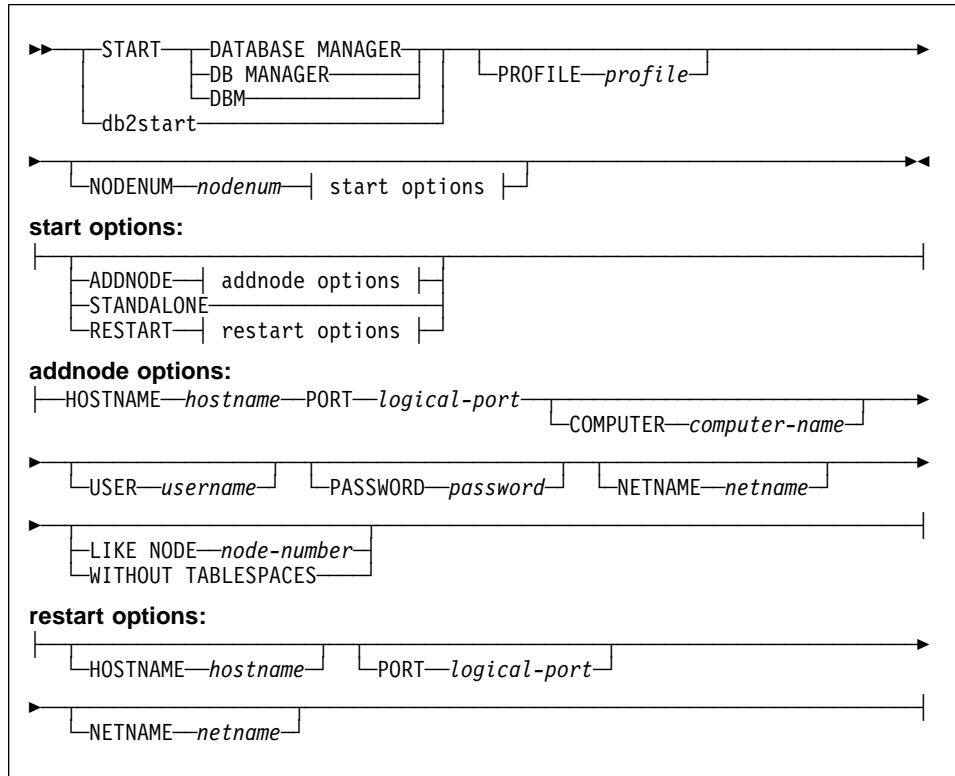
- sysadm*
- sysctrl*
- sysmaint*

Required Connection

None

START DATABASE MANAGER

Command Syntax



Command Parameters

Note: All of the following parameters are valid in an MPP environment only.

PROFILE *profile*

Specifies the name of the profile file to be executed at each node to define the DB2 environment. This file is executed before the nodes are started. The profile file must reside in the `sql1ib` directory of the instance owner.

Note: The environment variables in the profile file are not necessarily all defined in the user session.

NODENUM *nodenum*

Specifies the node to be started. If no other options are specified, a normal startup is done at this node.

Valid values are from 0 to 999 inclusive. If `ADDNODE` is not specified, the value must already exist in the `db2nodes.cfg` file of the instance owner. If no node number is specified, all nodes defined in the node configuration file are started.

START DATABASE MANAGER

ADDNODE

Specifies that the new node is added to the `db2nodes.cfg` file of the instance owner with the *hostname* and *logical-port* values.

Ensure that the combination of *hostname* and *logical-port* is unique.

The add node utility is executed internally to create all existing databases on the node being added. After a node is added, the `db2nodes.cfg` file is not updated with the new node until a **db2stop** is issued. The node is not part of the MPP system until the next **db2start** following the **db2stop**.

Note: When the database partitions are created on the new node, their configuration parameters are set to the default.

HOSTNAME *hostname*

With ADDNODE, specifies the host name to be added to the `db2nodes.cfg` file.

PORT *logical-port*

With ADDNODE, specifies the logical port to be added to the `db2nodes.cfg` file. Valid values are from 0 to 999.

COMPUTER *computer-name*

The computer name for the machine on which the new node is created. This parameter is mandatory on Windows NT, but is ignored on other operating systems.

USER *username*

The user name for the account on the new node. This parameter is mandatory on Windows NT, but is ignored on other operating systems.

PASSWORD *password*

The password for the account on the new node. This parameter is mandatory on Windows NT, but is ignored on other operating systems.

NETNAME *netname*

Specifies the *netname* to be added to the `db2nodes.cfg` file. If not specified, this parameter defaults to the value specified for *hostname*.

LIKE NODE *node-number*

Specifies that the containers for the temporary table spaces will be the same as the containers on the specified *node-number* for each database in the instance. The node specified must be a node that is already in the `db2nodes.cfg` file.

WITHOUT TABLESPACES

Specifies that containers for the temporary table spaces are not created for any of the databases. The ALTER TABLESPACE statement must be used to add temporary table

START DATABASE MANAGER

space containers to each database before the database can be used.

STANDALONE

Specifies that the node is to be started in stand-alone mode. FCM does not attempt to establish a connection to any other node. This option is used when adding a node.

RESTART

Starts the database manager after a failure. Other nodes are still operating, and this node attempts to connect to the others. If neither the *hostname* nor the *logical-port* parameter is specified, the database manager is restarted using the *hostname* and *logical-port* values specified in `db2nodes.cfg`. If either parameter is specified, the new values are sent to the other nodes when a connection is established. The `db2nodes.cfg` file is updated with this information.

HOSTNAME *hostname*

With RESTART, specifies the host name to be used to override that in the node configuration file.

PORT *logical-port*

With RESTART, specifies the logical port number to be used to override that in the node configuration file. If not specified, this parameter defaults to the *logical-port* value that corresponds to the *nodenum* value in the `db2nodes.cfg` file. Valid values are from 0 to 999.

NETNAME *netname*

Specifies the *netname* to override that specified in the `db2nodes.cfg` file. If not specified, this parameter defaults to the *netname* value that corresponds to the *nodenum* value in the `db2nodes.cfg` file.

Example

The following is sample output from **db2start** issued on a three node system with nodes 10, 20, and 30:

```
04-07-1997 10:33:05 10 0 SQL1063N DB2START processing was successful.
04-07-1997 10:33:07 20 0 SQL1063N DB2START processing was successful.
04-07-1997 10:33:07 30 0 SQL1063N DB2START processing was successful.
SQL1063N DB2START processing was successful.
```

Usage Notes

It is not necessary to issue this command on a client node. It is provided for compatibility with older clients, but it has no effect on the database manager.

Once started, the database manager instance runs until the user stops it, even if all application programs that were using it have ended.

If the database manager starts successfully, a successful completion message is sent to the standard output device. If an error occurs, processing stops, and an error

START DATABASE MANAGER

message is sent to the standard output device. In a multi-node environment, messages are returned on the node that issued the START DATABASE MANAGER command.

If no parameters are specified in a multi-node database environment, the database manager is started on all parallel nodes using the parameters specified in the node configuration file.

If a START DATABASE MANAGER command is in progress, ensure that the applicable nodes have started *before* issuing a request to the database.

The db2cshrc file is not supported and cannot be used to define the environment.

On UNIX platforms, the START DATABASE MANAGER command supports the SIGINT and SIGALRM signals. The SIGINT signal is issued if CTRL+C is pressed. The SIGALRM signal is issued if the value specified for the *start_stop_time* database manager configuration parameter is reached. If either signal occurs, all in-progress startups are interrupted and a message (SQL1044N for SIGINT and SQL6037N for SIGALRM) is returned from each interrupted node to the `$HOME/sqllib/log/db2start.timestamp.log` error log file. Nodes that are already started are not affected. If CTRL+C is pressed on a node that is starting, **db2stop** must be issued on that node before an attempt is made to start it again.

See Also

“ADD NODE” on page 89

“STOP DATABASE MANAGER” on page 395.

STOP DATABASE MANAGER

STOP DATABASE MANAGER

Stops the current database manager instance. Unless explicitly stopped, the database manager continues to be active. This command does not stop the database manager instance if any applications are connected to databases. If there are no database connections, but there are instance attachments, it forces the instance attachments and stops the database manager. This command also deactivates any outstanding database activations before stopping the database manager.

On an MPP system, this command stops the current database manager instance on a node or on all nodes. When it stops the database manager on all nodes, it uses the node configuration file `db2nodes.cfg` to obtain information about each node.

This command can also be used to drop a node from the `db2nodes.cfg` file (MPP systems only).

This command is not valid on a client.

Scope

By default, and in a multi-node environment, this command affects all nodes that are listed in the `db2nodes.cfg` file.

Authorization

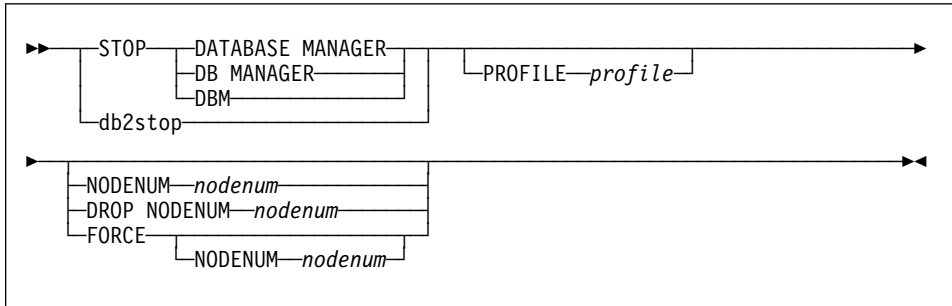
One of the following:

- `sysadm`
- `sysctrl`
- `sysmaint`

Required Connection

None

Command Syntax



STOP DATABASE MANAGER

Command Parameters

PROFILE *profile*

MPP only. Specifies the name of the profile file that was executed at startup to define the DB2 environment for those nodes that were started. If a profile for "START DATABASE MANAGER" on page 390 was specified, the same profile must be specified here. The profile file must reside in the `sql1lib` directory of the instance owner.

NODENUM *nodenum*

MPP only. Specifies the node to be stopped.

Valid values are from 0 to 999 inclusive, and must be in the `db2nodes.cfg` file. If no node number is specified, all nodes defined in the node configuration file are stopped.

DROP NODENUM *nodenum*

MPP only. Specifies the node to be dropped from the `db2nodes.cfg` file.

Before using this parameter, run "DROP NODE VERIFY" on page 159 to ensure that there is no user data on this node.

When this option is specified, all nodes in the `db2nodes.cfg` file are stopped.

FORCE

Specifies to use FORCE APPLICATION ALL when stopping the database manager at each node.

NODENUM *nodenum*

MPP only. Specifies the node (database partition server) to be stopped after all applications on that node have been forced to stop. If the FORCE option is used without this parameter, all applications on all nodes are forced before all the nodes are stopped.

Example

The following is sample output from **db2stop** issued on a three node system with nodes 10, 20, and 30:

```
04-07-1997 10:32:53 10 0 SQL1064N DB2STOP processing was successful.
04-07-1997 10:32:54 20 0 SQL1064N DB2STOP processing was successful.
04-07-1997 10:32:55 30 0 SQL1064N DB2STOP processing was successful.
SQL1064N DB2STOP processing was successful.
```

Usage Notes

It is not necessary to issue this command on a client node. It is provided for compatibility with older clients, but it has no effect on the database manager.

Once started, the database manager instance runs until the user stops it, even if all application programs that were using it have ended.

STOP DATABASE MANAGER

If the database manager is stopped, a successful completion message is sent to the standard output device. If an error occurs, processing stops, and an error message is sent to the standard output device.

If the database manager cannot be stopped because application programs are still connected to databases, use “FORCE APPLICATION” on page 169 to disconnect all users first, or reissue the STOP DATABASE MANAGER command with the FORCE option.

The following information currently applies to multi-node environments only:

- If no parameters are specified, the database manager is stopped on each node listed in the node configuration file. The `db2diag.log` file may contain messages to indicate that other nodes are shutting down.
- Any nodes added to the MPP system since the previous STOP DATABASE MANAGER command was issued will be updated in the `db2nodes.cfg` file.
- On UNIX platforms, this command supports the SIGALRM signal, which is issued if the value specified for the `start_stop_time` database manager configuration parameter is reached. If this signal occurs, all in-progress stops are interrupted, and message SQL6037N is returned from each interrupted node to the `$HOME/sql1lib/log/db2stop.timestamp.log` error log file. Nodes that are already stopped are not affected.
- The `db2cshrc` file is not supported and cannot be specified as the value for the PROFILE parameter.

Attention: The UNIX `kill` command should *not* be used to terminate the database manager because it will abruptly end database manager processes without controlled termination and cleanup processing.

See Also

“DEACTIVATE DATABASE” on page 149

“DROP NODE VERIFY” on page 159

“FORCE APPLICATION” on page 169

“START DATABASE MANAGER” on page 390.

TERMINATE

TERMINATE

Explicitly terminates the command line processor's back-end process. For more information about back-end and front-end processes, see “Command Line Processor Design” on page 79.

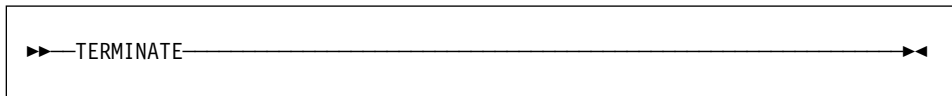
Authorization

None

Required Connection

None

Command Syntax



```
▶—TERMINATE—◀◀
```

Command Parameters

None

Usage Notes

If an application is connected to a database, or a process is in the middle of a unit of work, **TERMINATE** causes the database connection to be lost. An internal commit is then performed.

Although **TERMINATE** and **CONNECT RESET** both break the connection to a database, only **TERMINATE** results in termination of the back-end process.

It is recommended that **TERMINATE** be issued if **db2start** and **db2stop** were executed while the back-end process was active. This prevents the back-end process from maintaining an attachment to a database manager instance that is no longer available.

Back-end processes in MPP systems must also be terminated when the **DB2NODE** environment variable is updated in the session. This environment variable is used to specify the coordinator node number within an MPP multiple logical node configuration.

UNCATALOG DATABASE

Deletes a database entry from the system database directory.

Authorization

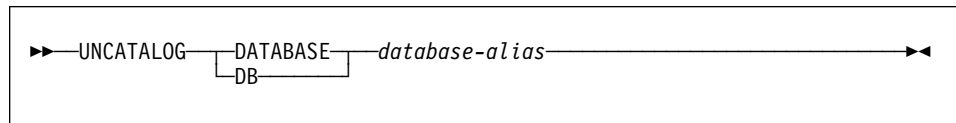
One of the following:

sysadm
sysctrl

Required Connection

None. Directory operations affect the local directory only.

Command Syntax



Command Parameters

DATABASE *database-alias*

Specifies the alias of the database to uncatalog.

Usage Notes

Only entries in the system database directory can be uncataloged. Entries in the local database directory can be deleted using “DROP DATABASE” on page 157.

To recatalog the database, use “CATALOG DATABASE” on page 118. To list the databases that are cataloged on a node, use “LIST DATABASE DIRECTORY” on page 240.

The authentication type of a database, used when communicating with a down-level server, can be changed by first uncataloging the database, and then cataloging it again with a different type.

Note: If directory caching is enabled (see the configuration parameter *dir_cache* in “GET DATABASE MANAGER CONFIGURATION” on page 188), database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use “TERMINATE” on page 398. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database. To

UNCATALOG DATABASE

refresh the directory cache for another application, stop and then restart that application.

UNCATALOG DCS DATABASE

Deletes an entry from the Database Connection Services (DCS) directory.

Authorization

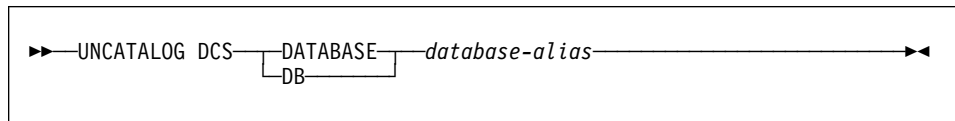
One of the following:

sysadm
sysctrl

Required Connection

None. Directory operations affect the local directory only.

Command Syntax



Command Parameters

DATABASE *database-alias*

Specifies the alias of the DCS database to uncatalog.

Usage Notes

DCS databases are also cataloged in the system database directory as remote databases that can be uncataloged using “UNCATALOG DATABASE” on page 399.

To recatalog a database in the DCS directory, use “CATALOG DCS DATABASE” on page 121. To list the DCS databases that are cataloged on a node, use “LIST DCS DIRECTORY” on page 247.

Note: If directory caching is enabled (see the configuration parameter *dir_cache* in “GET DATABASE MANAGER CONFIGURATION” on page 188), database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use “TERMINATE” on page 398. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database. To refresh the directory cache for another application, stop and then restart that application.

UNCATALOG NODE

UNCATALOG NODE

Deletes an entry from the node directory.

Authorization

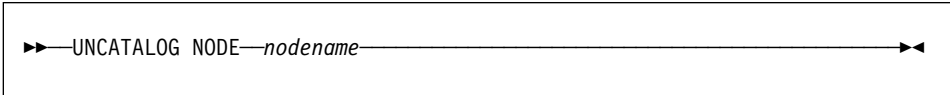
One of the following:

sysadm
sysctrl

Required Connection

None. Directory operations affect the local directory only.

Command Syntax



► UNCATALOG NODE *nodename* ◄

The diagram shows the command syntax for the UNCATALOG NODE command. It consists of a rectangular box containing the text '► UNCATALOG NODE *nodename* ◄'. A horizontal line with arrowheads at both ends extends from the text, indicating the range of the command and its parameter.

Command Parameters

NODE *nodename*

Specifies the node entry being uncataloged.

Usage Notes

UNCATALOG NODE can be executed on any type of node, but only the local directory is affected, even if there is an attachment to a remote instance, or a different local instance.

Note: If directory caching is enabled (see the configuration parameter *dir_cache* in “GET DATABASE MANAGER CONFIGURATION” on page 188), database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications may not be effective until the application has restarted.

To refresh the CLP's directory cache, use “TERMINATE” on page 398. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database. To refresh the directory cache for another application, stop and then restart that application.

See Also

“CATALOG APPC NODE” on page 111
“CATALOG APPCLU NODE” on page 114
“CATALOG APPN NODE” on page 116
“CATALOG IPX/SPX NODE” on page 126

UNCATALOG NODE

"CATALOG LOCAL NODE" on page 129
"CATALOG NETBIOS NODE" on page 133
"CATALOG NAMED PIPE NODE" on page 131
"CATALOG TCP/IP NODE" on page 136.

UNCATALOG ODBC DATA SOURCE

UNCATALOG ODBC DATA SOURCE

Uncatalogs a user or system ODBC data source.

A *data source*, in ODBC (Open Database Connectivity) terminology, is a user-defined name for a specific database or file system. That name is used to access the database or file system through ODBC APIs. On Windows NT and Windows 95, either user or system data sources can be uncataloged. A user data source is only visible to the user who cataloged it, whereas a system data source is visible to and can be used by all other users.

This command is available on Windows NT, Windows 95, and Windows 3.1 only.

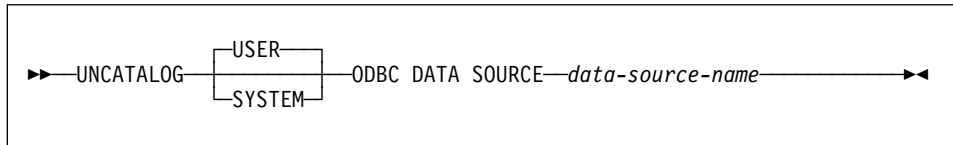
Authorization

None

Required Connection

None

Command Syntax



Command Parameters

USER

Uncatalog a user data source. This is the default if no keyword is specified.

SYSTEM

Uncatalog a system data source.

ODBC DATA SOURCE *data-source-name*

Specifies the name of the data source to be uncataloged. Maximum length is 32 characters.

See Also

“CATALOG ODBC DATA SOURCE” on page 135

“LIST ODBC DATA SOURCES” on page 264.

UPDATE ADMIN CONFIGURATION

Modifies individual entries in the database manager configuration file that are relevant to the DB2 Administration Server. The DB2 Administration Server is a special DB2 instance that enables remote administration of DB2 servers. The following database manager configuration parameters can be modified:

- AGENT_STACK_SZ
- AUTHENTICATION
- DIAGLEVEL
- DIAGPATH
- DISCOVER
- DISCOVER_COMM
- FILESERVER
- IPX_SOCKET
- NNAME
- OBJECTNAME
- QUERY_HEAP_SZ
- SYSADM_GROUP
- SYSCTRL_GROUP
- SYSMANT_GROUP
- TPNAME
- TRUST_ALLCLNTS
- TRUST_CLNTAUTH

Note: It is not recommended that the SVCENAME parameter, set by the installation program, be modified by the user. The administration server service name is set to use the DB2 registered TCP/IP port (523).

For more information about these parameters, see “GET DATABASE MANAGER CONFIGURATION” on page 188.

Scope

This command can be issued from any node listed in the `db2nodes.cfg` file. It affects all nodes that are listed in this file.

Authorization

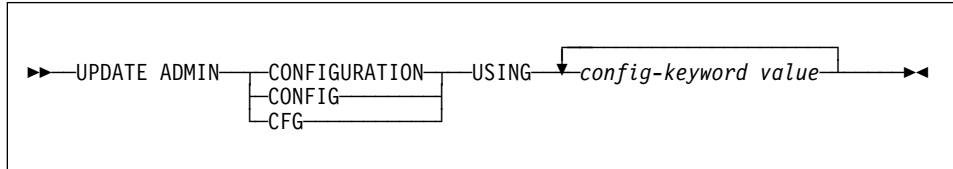
sysadm

Required Connection

None or instance. An instance attachment is not required to perform local DBM configuration operations, but is required to perform remote DBM configuration operations. To update the database manager configuration for a remote instance, it is necessary to first attach to that instance.

UPDATE ADMIN CONFIGURATION

Command Syntax



Command Parameters

USING *config-keyword value*

Specifies the admin configuration parameter to be updated.

Usage Notes

To view or print a list of the admin configuration parameters, use “GET ADMIN CONFIGURATION” on page 171.

To reset the admin configuration parameters to the recommended database manager defaults, use “RESET ADMIN CONFIGURATION” on page 352.

For more information about admin configuration parameters, see the *Administration Guide*.

The values of these parameters differ for each type of database node configured (server, client, or server with remote clients). See the *Administration Guide*, or one of the *Quick Beginnings* books for the ranges and the default values that can be set on each node type.

Changes to the database manager configuration file become effective only after they are loaded into memory. This occurs during execution of **db2start**.

If an error occurs, the database manager configuration file does not change.

The database manager configuration file cannot be updated if the checksum is invalid. This may occur if the database manager configuration file is changed without using the appropriate command. If this happens, the database manager must be reinstalled to reset the database manager configuration file.

See Also

“GET ADMIN CONFIGURATION” on page 171

“RESET ADMIN CONFIGURATION” on page 352.

UPDATE CLI CONFIGURATION

Updates the contents of a specified section in the `db2cli.ini` file.

The `db2cli.ini` file is used as the DB2 call level interface (CLI) configuration file. It contains various keywords and values that can be used to modify the behavior of the DB2 CLI and the applications using it. The file is divided into sections, each section corresponding to a database alias name. For more information about this file and the CLI/ODBC configuration keywords, see the *CLI Guide and Reference*.

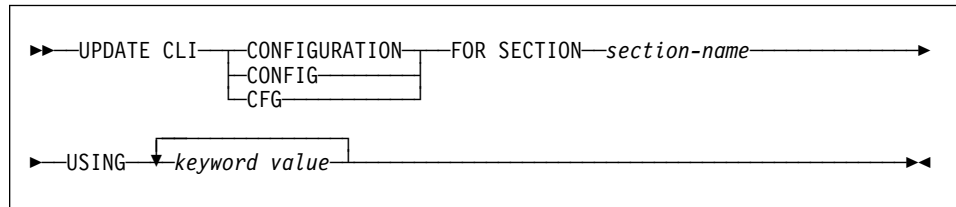
Authorization

`sysadm`

Required Connection

None

Command Syntax



Command Parameters

FOR SECTION *section-name*

Name of the section whose keywords are to be updated. If the specified section does not exist, a new section is created.

USING *keyword value*

Specifies the CLI/ODBC parameter to be updated.

Usage Notes

The section name and the keywords specified on this command are not case sensitive. However, the keyword values *are* case sensitive.

If a keyword value is a string containing single quotation marks or imbedded blanks, the entire string must be delimited by double quotation marks. For example:

```

db2 update cli cfg for section tstcli1x
    using TableType "'TABLE','VIEW','SYSTEM TABLE'"
  
```

UPDATE CLI CONFIGURATION

See Also

“GET CLI CONFIGURATION” on page 176.

UPDATE COMMAND OPTIONS

Sets one or more command options during an interactive session, or from a batch input file. The settings revert to system defaults (or the system default overrides in **DB2OPTIONS**) when the interactive session or batch input file ends.

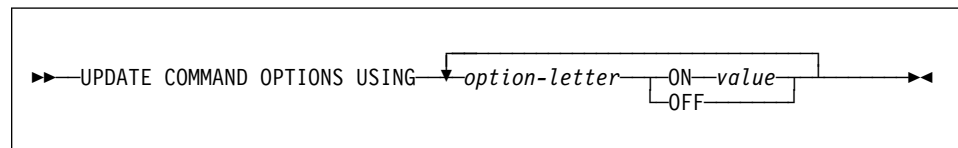
Authorization

None

Required Connection

None

Command Syntax



Command Parameters

USING *option-letter*

The following option-letters can be set:

- a** Display SQLCA
- c** Auto-commit SQL statements
- e** Display SQLCODE/SQLSTATE
- l** Log commands in a history file
- o** Display to standard output
- p** Display DB2 interactive prompt
- r** Save output report to a file
- s** Stop execution on command error
- v** Echo current command
- w** Show SQL statement warning messages
- z** Redirect all output to a file.

ON *value*

The e, l, r, and z options require a value if they are turned on. For the e option, *value* can be c to display the SQLCODE, or s to display the SQLSTATE. For the l, r, and z options, *value* represents the name to be used for the history file or the report file. No other options accept a value.

UPDATE COMMAND OPTIONS

Usage Notes

These settings override system defaults, settings in **DB2OPTIONS**, and options specified using the command line option flags.

The file input option (-f) and the statement termination option (-t) cannot be updated using this command.

To view the current option settings, use “LIST COMMAND OPTIONS” on page 238.

For detailed information about these options, see “Command Line Processor Invocation and Options” on page 69.

UPDATE DATABASE CONFIGURATION

UPDATE DATABASE CONFIGURATION

Modifies individual entries in a specific database configuration file.

A database configuration file resides on every node on which the database has been created.

Scope

This command only affects the node on which it is executed.

Authorization

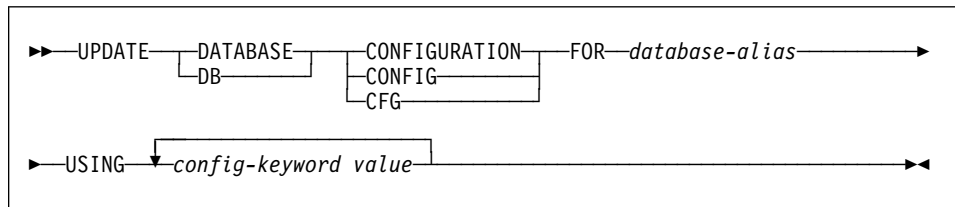
One of the following:

sysadm
sysctrl
sysmaint

Required Connection

Instance. An explicit attachment is not required. If the database is listed as remote, an instance attachment to the remote node is established for the duration of the command.

Command Syntax



Command Parameters

FOR *database-alias*

Specifies the alias of the database whose configuration is to be updated.

USING *config-keyword value*

Specifies the database configuration parameter to be updated. For a brief description of configurable parameters, see “GET DATABASE CONFIGURATION” on page 179.

Usage Notes

To view or print a list of the database configuration parameters, use “GET DATABASE CONFIGURATION” on page 179.

To reset the database configuration parameters to the recommended database manager defaults, use “RESET DATABASE CONFIGURATION” on page 354.

UPDATE DATABASE CONFIGURATION

For more information about DB2's configuration parameters, see the *Administration Guide*.

The values of these parameters differ for each type of database node configured (server, client, or server with remote clients). See the *Administration Guide* for the ranges and the default values that can be set on each node type.

Not all parameters can be updated.

Changes to the database configuration file become effective only after they are loaded into memory. All applications must disconnect from the database before this can occur.

If an error occurs, the database configuration file does not change.

The database configuration file cannot be updated if the checksum is invalid. This may occur if the database configuration file is changed without using the appropriate command. If this happens, the database must be restored to reset the database configuration file.

See Also

"GET DATABASE CONFIGURATION" on page 179

"RESET DATABASE CONFIGURATION" on page 354.

UPDATE DATABASE MANAGER CONFIGURATION

UPDATE DATABASE MANAGER CONFIGURATION

Modifies individual entries in the database manager configuration file.

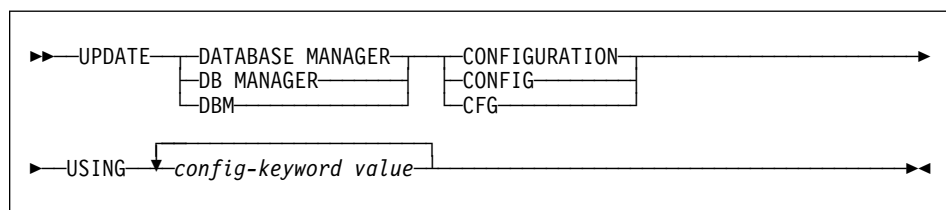
Authorization

sysadm

Required Connection

None or instance. An instance attachment is not required to perform local DBM configuration operations, but is required to perform remote DBM configuration operations. To update the database manager configuration for a remote instance, it is necessary to first attach to that instance.

Command Syntax



Command Parameters

USING *config-keyword value*

Specifies the database manager configuration parameter to be updated. For a brief description of configurable parameters, see “GET DATABASE MANAGER CONFIGURATION” on page 188.

Usage Notes

To view or print a list of the database manager configuration parameters, use “GET DATABASE MANAGER CONFIGURATION” on page 188.

To reset the database manager configuration parameters to the recommended database manager defaults, use “RESET DATABASE MANAGER CONFIGURATION” on page 356.

For more information about database manager configuration parameters, see the *Administration Guide*.

The values of these parameters differ for each type of database node configured (server, client, or server with remote clients). See the *Administration Guide* for the ranges and the default values that can be set on each node type.

Not all parameters can be updated.

UPDATE DATABASE MANAGER CONFIGURATION

Changes to the database manager configuration file become effective only after they are loaded into memory. For a server configuration parameter, this occurs during execution of **db2start**. For a client configuration parameter, this occurs when the application is restarted. If the client is the command line processor, it is necessary to invoke "TERMINATE" on page 398.

If an error occurs, the database manager configuration file does not change.

The database manager configuration file cannot be updated if the checksum is invalid. This may occur if the database manager configuration file is changed without using the appropriate command. If this happens, the database manager must be reinstalled to reset the database manager configuration file.

See Also

"GET DATABASE MANAGER CONFIGURATION" on page 188

"RESET DATABASE MANAGER CONFIGURATION" on page 356.

UPDATE MONITOR SWITCHES

Turns one or more database monitor recording switches on or off. When the database manager starts, the settings of the six switches are determined by the *dft_mon* database manager configuration parameters (see “GET DATABASE MANAGER CONFIGURATION” on page 188).

The database monitor records a base set of information at all times. Users who require more than this basic information can turn on the appropriate switches, but at a cost to system performance. The amount of information available in output from “GET SNAPSHOT” on page 203 reflects which, if any, switches are on.

Authorization

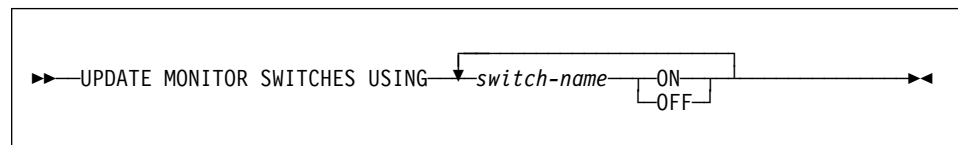
One of the following:

sysadm
sysctrl
sysmaint

Required Connection

Instance. To update the monitor switches at a remote instance (or a different local instance), it is necessary to first attach to that instance.

Command Syntax



Command Parameters

USING *switch-name*

The following switch names are available:

BUFFERPOOL	Buffer pool activity information
LOCK	Lock information
SORT	Sorting information
STATEMENT	SQL statement information
TABLE	Table activity information
UOW	Unit of work information.

UPDATE MONITOR SWITCHES

Usage Notes

Information is collected by the database manager only after a switch is turned on. The switches remain set until **db2stop** is issued. To clear the information related to a particular switch, set the switch off, then on.

Updating switches in one application does not affect other applications.

To view the switch settings, use “GET MONITOR SWITCHES” on page 201.

UPDATE RECOVERY HISTORY FILE

UPDATE RECOVERY HISTORY FILE

Updates the location, device type, or comment in a recovery history file entry.

Authorization

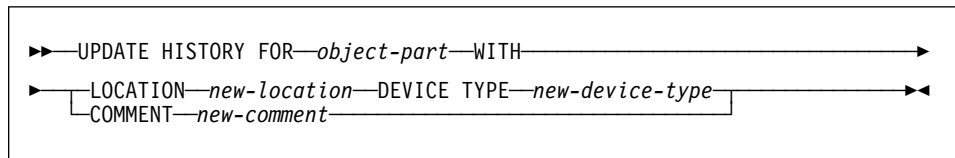
One of the following:

sysadm
sysctrl
sysmaint
dbadm

Required Connection

Database

Command Syntax



Command Parameters

FOR *object-part*

Specifies the identifier for the backup or copy image. It is a time stamp with a sequence number from 001 to 999.

LOCATION *new-location*

Specifies the new physical location of a backup. The interpretation of this parameter depends on the device type.

DEVICE TYPE *new-device-type*

Specifies a new device type for storing the backup. Valid device types are:

D Disk
K Diskette
T Tape
A ADSM
U User exit
O Other.

COMMENT *new-comment*

Specifies a new comment to describe the entry.

Example

To update the history file entry for the full database backup taken on April 13, 1997 at 10:00 a.m., enter:

```
db2 update history for 19970413100000001 with  
location /backup/dbbackup.1 device type d
```

UPDATE RECOVERY HISTORY FILE

Usage Notes

The recovery history file is used for record keeping purposes only. It is not used during database recovery.

See Also

“PRUNE HISTORY” on page 325.

Chapter 4. Using Command Line SQL Statements

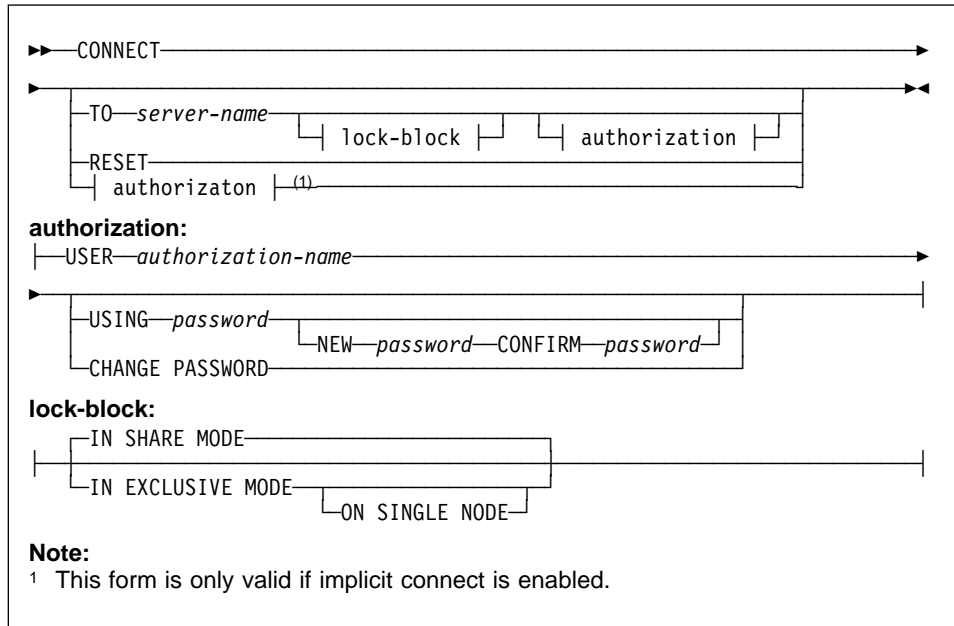
This section provides information about using Structured Query Language (SQL) statements from the command line. These statements can be executed directly from an operating system command prompt, and can be used to define and manipulate information stored in a database table, index, or view in much the same way as if the commands were written into an application program. Information can be added, deleted, or updated, and reports can be generated from the contents of tables.

All SQL statements that can be executed through the command line processor are listed in the CLP column of Table 9 on page 423. The syntax of all the SQL statements, whether executed from the command line or embedded in a source program, is described in the *SQL Reference*. The syntax of many embedded SQL statements and CLP SQL statements is identical. However, host variables, parameter markers, descriptor names, and statement names are applicable only to embedded SQL. The syntax of CLOSE, CONNECT, DECLARE CURSOR, FETCH, OPEN, and SELECT *does* depend on whether these statements are embedded or executed through the CLP. The CLP syntax of these statements is provided below:

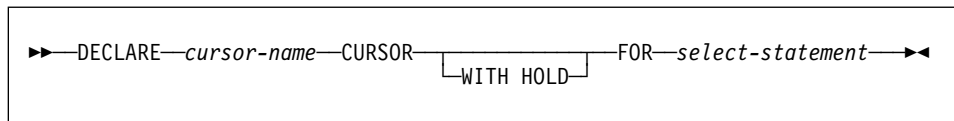
CLOSE

▶▶—CLOSE— <i>cursor-name</i> ————▶▶

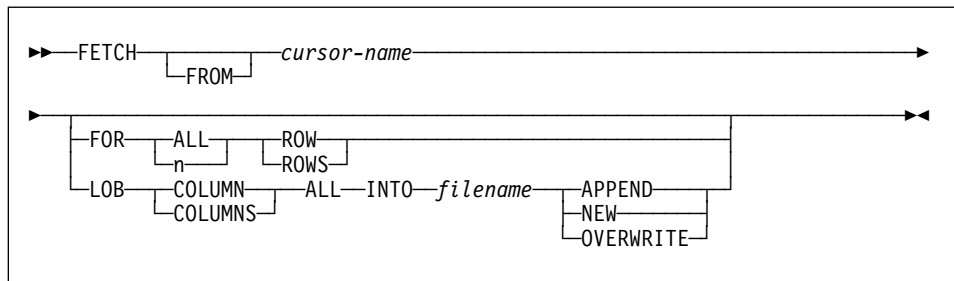
CONNECT



DECLARE CURSOR



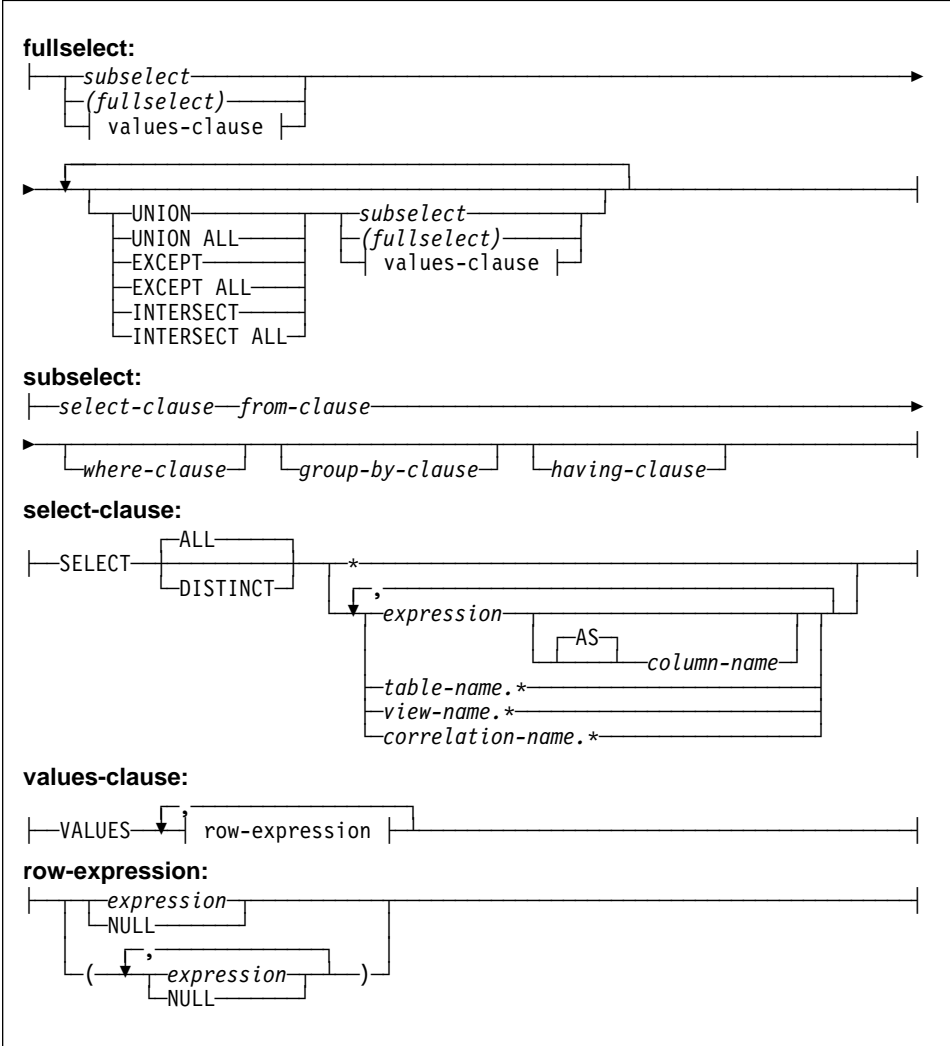
FETCH



OPEN



SELECT



Notes:

1. The CLP version of CONNECT permits the user to change the password, using the following parameters:

NEW *password*

Specifies the new password that is to be assigned to the user name. Passwords can be up to 18 characters in length. The system on which the password will be changed depends on how user authentication has been set up.

CONFIRM *password*

A string that must be identical to the new password. This parameter is used to catch entry errors.

CHANGE PASSWORD

If this option is specified, the user is prompted for the current password, a new password, and for confirmation of the new password. Passwords are not displayed at entry.

2. When FETCH or SELECT is issued through the command line processor, decimal and floating-point numbers are displayed with the country's decimal delimiter, that is, a period (.) in the U.S., Canada, and the U.K.; a comma (,) in most other countries. However, when INSERT, UPDATE, and other SQL statements are issued through the command line processor to update tables, a period must be used as the decimal delimiter, even in countries that use a comma for that purpose.
3. When FETCH or SELECT is issued through the command line processor, null values are typically displayed as a hyphen (-). For databases configured with DFT_SQLMATHWARN YES, expressions that result in an arithmetic error are processed as null values. Such arithmetic error nulls are displayed as a plus (+).

For example, create and populate table t1 as follows:

```
create table t1 (i1 int , i2 int);
insert into t1 values (1,1),(2,0),(3,null);
```

The statement: select i1/i2 from t1 generates the following result:

```
1
---
1
+
-
3 records selected
```

4. A new LOB option has been added to FETCH. If the LOB clause is specified, only the next row is fetched:
 - Each LOB column value is fetched into a file with the name *filename.xxx*, where *filename* is specified in the LOB clause, and *xxx* is a file extension from 001 to 999 (001 is the first LOB column in the select list of the corresponding DECLARE CURSOR statement, 002 is the second LOB column, and 999 is the

999th column). The maximum number of LOB columns that can be fetched into files is 999.

- Names of the files containing the data are displayed in the LOB columns.
5. When SELECT is issued through the command line processor to query tables containing LOB columns, each LOB column is truncated to 4KB in the output.
 6. The command line processor displays BLOB columns in hexadecimal representation.

Change the way that the CLP displays data (when querying databases using SQL statements through the CLP) by rebinding the CLP bind files against the database being queried. For example, to display date and time in ISO format, do the following:

1. Create a text file containing the names of the CLP bind files. This file is used as the list file for binding multiple files with one BIND command. In this example the file is named `clp.lst`, and its contents are:

```
db2clpcs.bnd +
db2clpr.r.bnd +
db2clpur.bnd +
db2clprs.bnd +
db2clpns.bnd
```

2. Connect to the database.
3. Issue the following command:

```
db2 bind @clp.lst collection nullid datetime iso
```

For detailed information about the command line processor, see Chapter 2, “Command Line Processor (CLP)” on page 69. For more information about the syntax of SQL statements and the function provided by SQL statements, see the *SQL Reference*. For information about reading syntax diagrams, see Appendix A, “How to Read the Syntax Diagrams” on page 427.

Table 9 (Page 1 of 3). SQL Statements (DB2 Universal Database)

SQL Statement	Dynamic ¹	Command Line Processor (CLP)	Call Level Interface ³ (CLI)
ALTER { BUFFERPOOL, NODEGROUP, TABLE, TABLESPACE, TYPE, VIEW }	X	X	X
BEGIN DECLARE SECTION ²			
CALL			X ⁴
CLOSE		X	SQLCloseCursor(), SQLFreeStmt()
COMMENT ON	X	X	X
COMMIT	X	X	SQLEndTran, SQLTransact()
Compound SQL			X ⁴

Table 9 (Page 2 of 3). SQL Statements (DB2 Universal Database)

SQL Statement	Dynamic ¹	Command Line Processor (CLP)	Call Level Interface ³ (CLI)
CONNECT (Type 1)		X	SQLBrowseConnect(), SQLConnect(), SQLDriverConnect()
CONNECT (Type 2)		X	SQLBrowseConnect(), SQLConnect(), SQLDriverConnect()
CREATE { ALIAS, BUFFERPOOL, DISTINCT TYPE, EVENT MONITOR, FUNCTION, INDEX, NODEGROUP, PROCEDURE, SCHEMA, TABLE, TABLESPACE, TRIGGER, TYPE, VIEW }	X	X	X
DECLARE CURSOR ²		X	SQLAllocStmt()
DELETE	X	X	X
DESCRIBE ⁸		X	SQLColAttributes(), SQLDescribeCol(), SQLDescribeParam() ⁶
DISCONNECT		X	SQLDisconnect()
DROP	X	X	X
END DECLARE SECTION ²			
EXECUTE			SQLExecute()
EXECUTE IMMEDIATE			SQLExecDirect()
EXPLAIN	X	X	X
FETCH		X	SQLExtendedFetch() ⁷ , SQLFetch(), SQLFetchScroll() ⁷
FREE LOCATOR			X ⁴
GRANT	X	X	X
INCLUDE ²			
INSERT	X	X	X
LOCK TABLE	X	X	X
OPEN		X	SQLExecute(), SQLExecDirect()
PREPARE			SQLPrepare()
RELEASE		X	
RENAME TABLE	X	X	X
REVOKE	X	X	X
ROLLBACK	X	X	SQLEndTran(), SQLTransact()
select-statement	X	X	X
SELECT INTO			
SET CONNECTION		X	SQLSetConnection()

Table 9 (Page 3 of 3). SQL Statements (DB2 Universal Database)

SQL Statement	Dynamic ¹	Command Line Processor (CLP)	Call Level Interface ³ (CLI)
SET CONSTRAINTS	X	X	X
SET CURRENT DEGREE	X	X	X
SET CURRENT EXPLAIN MODE	X	X	X, SQLSetConnectAttr()
SET CURRENT EXPLAIN SNAPSHOT	X	X	X, SQLSetConnectAttr()
SET CURRENT FUNCTION PATH	X	X	X
SET CURRENT PACKAGESET			
SET CURRENT QUERY OPTIMIZATION	X	X	X
SET EVENT MONITOR STATE	X	X	X
SET transition-variable ⁵	X	X	X
SIGNAL SQLSTATE ⁵	X	X	X
UPDATE	X	X	X
VALUES INTO			
WHENEVER ²			

Notes:

1. You can code all statements in this list as static SQL, but only those marked with X as dynamic SQL.
2. You cannot execute this statement.
3. An X indicates that you can execute this statement using either `SQLExecDirect()` or `SQLPrepare()` and `SQLExecute()`. If there is an equivalent DB2 CLI function, the function name is listed.
4. Although this statement is not dynamic, with DB2 CLI you can specify this statement when calling either `SQLExecDirect()`, or `SQLPrepare()` and `SQLExecute()`.
5. You can only use this within `CREATE TRIGGER` statements.
6. You can only use the SQL `DESCRIBE` statement to describe output, whereas with DB2 CLI you can also describe input (using the `SQLDescribeParam()` function).
7. You can only use the SQL `FETCH` statement to fetch one row at a time in one direction, whereas with the DB2 CLI `SQLExtendedFetch()` and `SQLFetchScroll()` functions, you can fetch into arrays. Furthermore, you can fetch in any direction, and at any position in the result set.
8. The `DESCRIBE SQL` statement has a different syntax than that of the CLP `DESCRIBE` command. For information on the `DESCRIBE SQL` statement, refer to the *SQL Reference*. For information on the `DESCRIBE CLP` command, refer to the *Command Reference*.

Appendix A. How to Read the Syntax Diagrams

A syntax diagram shows how a command should be specified so that the operating system can correctly interpret what is typed.

Read a syntax diagram from left to right, and from top to bottom, following the horizontal line (the main path). If the line ends with an arrowhead, the command syntax is continued, and the next line starts with an arrowhead. A vertical bar marks the end of the command syntax.

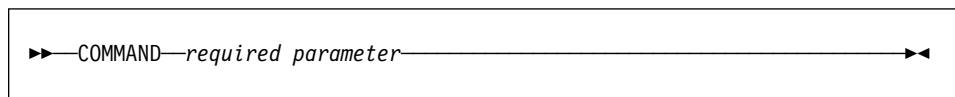
When typing information from a syntax diagram, be sure to include punctuation, such as quotation marks and equal signs.

Parameters are classified as keywords or variables:

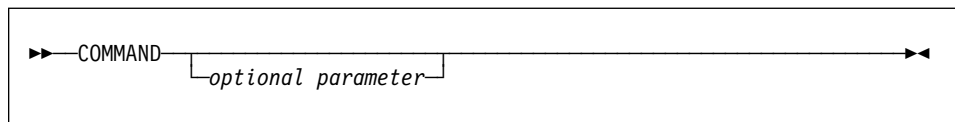
- Keywords represent constants, and are shown in uppercase letters; at the command prompt, however, keywords can be entered in upper, lower, or mixed case. A command name is an example of a keyword.
- Variables represent names or values that are supplied by the user, and are shown in lowercase letters; at the command prompt, however, variables can be entered in upper, lower, or mixed case, unless case restrictions are explicitly stated. A file name is an example of a variable.

A parameter can be a combination of a keyword and a variable.

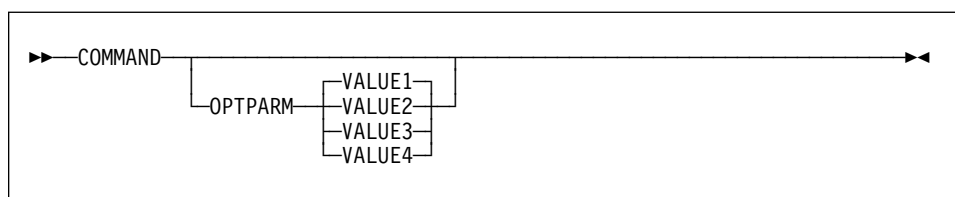
Required parameters are displayed on the main path:



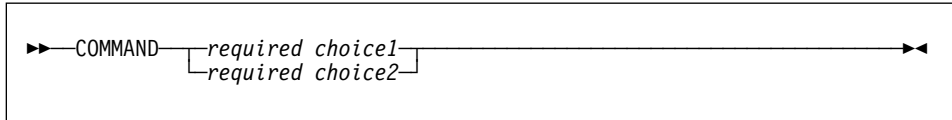
Optional parameters are displayed below the main path:



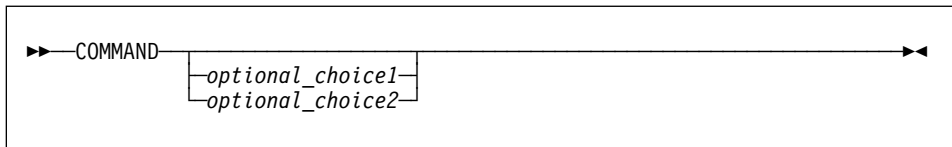
A parameter's default value is displayed above the path:



A stack of parameters, with the first parameter displayed on the main path, indicates that one of the parameters must be selected:

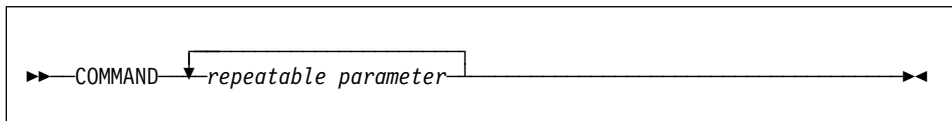


A stack of parameters, with the first parameter displayed below the main path, indicates that one of the parameters can be selected:

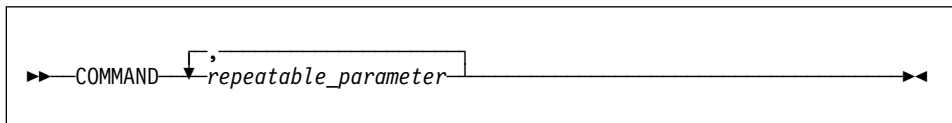


An arrow returning to the left, above the path, indicates that items can be repeated in accordance with the following conventions:

- If the arrow is uninterrupted, the item can be repeated in a list with the items separated by blank spaces:

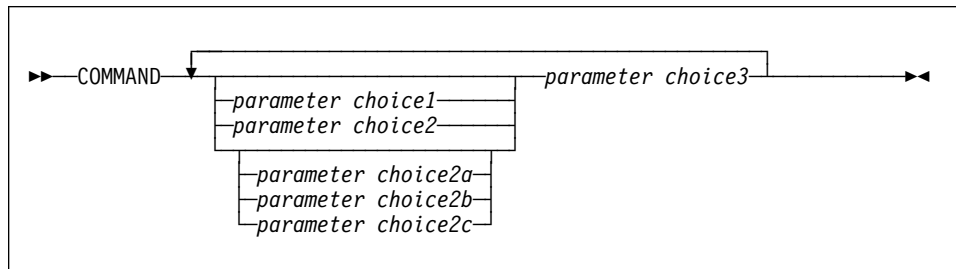


- If the arrow contains a comma, the item can be repeated in a list with the items separated by commas:

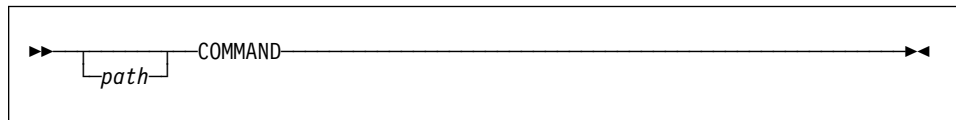


Items from parameter stacks can be repeated in accordance with the stack conventions for required and optional parameters discussed previously.

Some syntax diagrams contain parameter stacks within other parameter stacks. Items from stacks can only be repeated in accordance with the conventions discussed previously. That is, if an inner stack does not have a repeat arrow above it, but an outer stack does, only one parameter from the inner stack can be chosen and combined with any parameter from the outer stack, and that combination can be repeated. For example, the following diagram shows that one could combine parameter *choice2a* with parameter *choice2*, and then repeat that combination again (*choice2* plus *choice2a*):

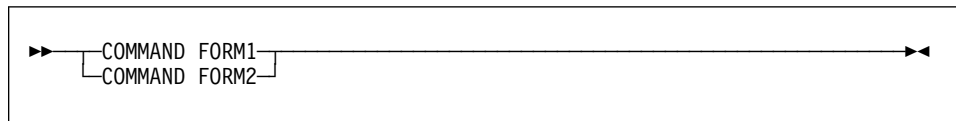


Some commands are preceded by an optional path parameter:



If this parameter is not supplied, the system searches the current directory for the command. If it cannot find the command, the system continues searching for the command in all the directories on the paths listed in the `.profile`.

Some commands have syntactical variants that are functionally equivalent:



Appendix B. Naming Conventions

This section provides information about the conventions that apply when naming database manager objects, such as databases and tables, and authentication IDs.

- Character strings that represent names of database manager objects can contain any of the following: a-z, A-Z, 0-9, @, #, and \$.
- The first character in the string must be an alphabetic character, @, #, or \$; it cannot be a number or the letter sequences SYS, DBM, or IBM.
- Unless otherwise noted, names can be entered in lowercase letters; however, the database manager processes them as if they were uppercase.

The exception to this is character strings that represent names under the systems network architecture (SNA). Many values, such as logical unit names (partner_lu and local_lu), are case sensitive. The name must be entered exactly as it appears in the SNA definitions that correspond to those terms.

- A database name or database alias is a unique character string containing from one to eight letters, numbers, or keyboard characters from the set described above.

Databases are cataloged in the system and local database directories by their aliases in one field, and their original name in another. For most functions, the database manager uses the name entered in the alias field of the database directories. (The exceptions are CHANGE DATABASE COMMENT and CREATE DATABASE, where a directory path must be specified.)

- The long identifier or alias for a database table or view, and the name of a column within a table or a view, are unique character strings 1 to 18 characters in length.

A fully qualified table name consists of the *schema.tablename*. The schema is the unique user ID under which the table was created.

- Authentication IDs (both user IDs and group IDs) cannot exceed eight characters in length.
- Local aliases for remote nodes that are to be cataloged in the node directory cannot exceed eight characters in length.

For more information about naming conventions, see the *Administration Guide*.

Appendix C. IMPORT/EXPORT/LOAD Utility File Formats

Three operating system file formats supported by the database manager for IMPORT, EXPORT, and LOAD are described. They are:

- DEL** Delimited ASCII, for exchange with many database managers and file managers.
- PC/IXF** PC version of IXF, an exchange format used by the database manager.
- ASC** Non-delimited ASCII (IMPORT only).

Note: Throughout this section, all comments that pertain to IMPORT are also applicable to LOAD.

Delimited ASCII (DEL) File Format

A Delimited ASCII (DEL) file is a sequential ASCII file with row and column delimiters. Each DEL file is a stream of ASCII characters consisting of cell values ordered by row, and then by column. Rows in the data stream are separated by row delimiters; within each row, individual cell values are separated by column delimiters.

The following table describes the format of DEL files that can be imported into, or that can be generated as the result of an export action.

<pre> DEL file ::= Row 1 data Row delimiter Row 2 data Row delimiter . . . Row n data Optional row delimiter Row i data ::= Cell value(i,1) Column delimiter Cell value(i,2) Column delimiter . . . Cell value(i,m) Row delimiter ::= ASCII line feed sequence^a Column delimiter ::= Default value ASCII comma (,)^b </pre>

Delimited ASCII (DEL) File Format

```
Cell value(i,j) ::= Leading spaces
                  || ASCII representation of a numeric value
                  || (integer, decimal, or float)
                  || Character string enclosed by character string delimiters
                  || Character string
                  || Trailing spaces

Character string delimiter ::= Default value ASCII double quotation
                              marks (")c

End-of-file character ::= Hex '1A' (OS/2 or the Windows operating system only)

ASCII representation of a numeric valued ::= Optional sign '+' or '-'
      || 1 to 31 decimal digits with an optional decimal point before,
      || after, or between two digits
      || Optional exponent

Exponent ::= Character 'E' or 'e'
      || Optional sign '+' or '-'
      || 1 to 3 decimal digits with no decimal point

Decimal digit ::= Any one of the characters '0', '1', ... '9'

Decimal point ::= Default value ASCII period (.)e
```

^a The record delimiter is assumed to be a new line character, ASCII x0A. Data generated on OS/2 or the Windows operating system can use the carriage return/line feed 2-byte standard of 0x0D0A. Data in EBCDIC code pages should use the EBCDIC LF character (0x25) as the record delimiter (EBCDIC data can be loaded using the CODEPAGE option on the LOAD command).

^b The column delimiter can be specified with the COLDEL option.

^c The character string delimiter can be specified with the CHARDEL option.

Note: The priority of delimiters is:

1. Record delimiter
2. Character delimiter
3. Column delimiter

^d If the ASCII representation of a numeric value contains an exponent, it is a FLOAT constant. If it has a decimal point but no exponent, it is a DECIMAL constant. If it has no decimal point and no exponent, it is an INTEGER constant.

^e The decimal point character can be specified with the DECPT option.

Sample DEL File

Following is an example of a DEL file. Each line ends with a line feed sequence (on OS/2 or the Windows operating system, each line ends with a carriage return/line feed sequence).

```
"Smith, Bob",4973,15.46
"Jones, Bill",12345,16.34
"Williams, Sam",452,193.78
```

Delimited ASCII (DEL) File Format

The following example illustrates the use of non-delimited character strings. The column delimiter has been changed to a semicolon, because the character data contains a comma.

```
Smith, Bob;4973;15.46
Jones, Bill;12345;16.34
Williams, Sam;452;193.78
```

Notes:

1. A space (X'20') is never a valid delimiter.
2. Spaces that precede the first character, or that follow the last character of a cell value, are discarded during import. Spaces that are embedded in a cell value are not discarded.
3. A period (.) is not a valid character string delimiter, because it conflicts with periods in time stamp values.
4. For pure DBCS (graphic), mixed DBCS, and EUC, delimiters are restricted to the range of x00 to x3F, inclusive.
5. For DEL data specified in an EBCDIC code page, the delimiters may not coincide with the shift-in and shift-out DBCS characters.
6. On OS/2 or the Windows operating system, the first occurrence of an end-of-file character (X'1A') that is not within character delimiters indicates the end-of-file. Any subsequent data is not imported.
7. If the first character of a cell value is the character string delimiter, the cell value is imported as a *delimited* character string. The string is terminated by the next occurrence of the character string delimiter; characters that follow this second delimiter but precede the next column delimiter are discarded during import.

Import of a delimited character string which contains a character string delimiter thus produces erroneous results. An attempt to export character data containing a character string delimiter causes a warning message. The import and export utilities permit the user to change the character string delimiter, thereby offering flexibility in resolving such problems.

A *non-delimited* character string is a cell value whose first character is not the character string delimiter. A non-delimited character string is terminated by the next occurrence of a column or a row delimiter, and thus may contain the character string delimiter. However, a non-delimited character string should not contain a column delimiter, a carriage return (on OS/2 or the Windows operating system), or the line feed sequence.

8. A null value is indicated by the absence of a cell value where one would normally occur, or by a string of spaces.
9. Since some products restrict character fields to 254 or 255 bytes, the export utility generates a warning message whenever a character column of maximum length greater than 254 bytes is selected for export. The import utility accommodates

Delimited ASCII (DEL) File Format

fields that are as long as the longest LONG VARCHAR and LONG VARGRAPHIC columns.

DEL Data Type Descriptions

Table 10 (Page 1 of 3). Acceptable Data Type Forms for the DEL File Format

Data Type	Form in Files Created by the Export Utility	Form Acceptable to the Import Utility
BIGINT	An INTEGER constant in the range -9 223 372 036 854 775 808 to 9 223 372 036 854 775 807.	ASCII representation of a numeric value in the range -9 223 372 036 854 775 808 to 9 223 372 036 854 775 807. Decimal and float numbers are truncated to integer values.
BLOB, CLOB	Character data enclosed by character delimiters (for example, double quotation marks).	A delimited or non-delimited character string. The character string is used as the database column value.
BLOB_FILE, CLOB_FILE	The character data for each BLOB/CLOB column is stored in individual files, and the file name is enclosed by character delimiters.	The delimited or non-delimited name of the file that holds the data.
CHAR	Character data enclosed by character delimiters (for example, double quotation marks).	A delimited or non-delimited character string. The character string is truncated or padded with spaces (X'20'), if necessary, to match the width of the database column.
DATE	<i>yyyymmdd</i> (year month day) with no character delimiters. For example: 19931029 Alternatively, the DATESISO option can be used to specify that all date values are to be exported in ISO format.	A delimited or non-delimited character string containing a date value in an ISO format consistent with the country code of the target database, or a non-delimited character string of the form <i>yyyymmdd</i> .
DBCLOB (DBCS only)	Graphic data is exported as a delimited character string.	A delimited or non-delimited character string, an even number of bytes in length. The character string is used as the database column value.

Delimited ASCII (DEL) File Format

Table 10 (Page 2 of 3). Acceptable Data Type Forms for the DEL File Format

Data Type	Form in Files Created by the Export Utility	Form Acceptable to the Import Utility
DBCLOB_FILE (DBCS only)	The character data for each DBCLOB column is stored in individual files, and the file name is enclosed by character delimiters.	The delimited or non-delimited name of the file that holds the data.
DECIMAL	A DECIMAL constant with the precision and scale of the field being exported. The DECPLUSBLANK option can be used to specify that positive decimal values are to be prefixed with a blank space instead of a plus sign (+).	ASCII representation of a numeric value that does not overflow the range of the database column into which the field is being imported. If the input value has more digits after the decimal point than can be accommodated by the database column, the excess digits are truncated.
FLOAT(long)	A FLOAT constant in the range -10E307 to 10E307.	ASCII representation of a numeric value in the range -10E307 to 10E307.
GRAPHIC (DBCS only)	Graphic data is exported as a delimited character string.	A delimited or non-delimited character string, an even number of bytes in length. The character string is truncated or padded with double-byte spaces (for example, X'8140'), if necessary, to match the width of the database column.
INTEGER	An INTEGER constant in the range -2 147 483 648 to 2 147 483 647.	ASCII representation of a numeric value in the range -2 147 483 648 to 2 147 483 647. Decimal and float numbers are truncated to integer values.
LONG VARCHAR	Character data enclosed by character delimiters (for example, double quotation marks).	A delimited or non-delimited character string. The character string is used as the database column value.
LONG VARGRAPHIC (DBCS only)	Graphic data is exported as a delimited character string.	A delimited or non-delimited character string, an even number of bytes in length. The character string is used as the database column value.

PC Version of IXF File Format

<i>Table 10 (Page 3 of 3). Acceptable Data Type Forms for the DEL File Format</i>		
Data Type	Form in Files Created by the Export Utility	Form Acceptable to the Import Utility
SMALLINT	An INTEGER constant in the range -32768 to 32767.	ASCII representation of a numeric value in the range -32768 to 32767. Decimal and float numbers are truncated to integer values.
TIME	<i>hh.mm.ss</i> (hour minutes seconds). A time value in ISO format enclosed by character delimiters. For example: "09.39.43"	A delimited or non-delimited character string containing a time value in a format consistent with the country code of the target database.
TIMESTAMP	<i>yyyy-mm-dd-hh.mm.ss.nnnnnn</i> (year month day hour minutes seconds microseconds). A character string representing a date and time enclosed by character delimiters.	A delimited or non-delimited character string containing a time stamp value acceptable for storage in a database.
VARCHAR	Character data enclosed by character delimiters (for example, double quotation marks).	A delimited or non-delimited character string. The character string is truncated, if necessary, to match the maximum width of the database column.
VARGRAPHIC (DBCS only)	Graphic data is exported as a delimited character string.	A delimited or non-delimited character string, an even number of bytes in length. The character string is truncated, if necessary, to match the maximum width of the database column.

PC Version of IXF File Format

The PC version of IXF (PC/IXF) file format is a database manager adaptation of the Integration Exchange Format (IXF) data interchange architecture. The IXF architecture was specifically designed to enable the exchange of relational database structures and data. The PC/IXF architecture allows the database manager to export a database without having to anticipate the requirements and idiosyncrasies of a receiving product. Similarly, a product importing a PC/IXF file need only understand the PC/IXF architecture; the characteristics of the product which exported the file are not relevant. The PC/IXF file architecture maintains the independence of both the exporting and the importing database systems.

The IXF architecture is a generic relational database exchange format that supports a rich set of relational data types, including some types that may not be supported by specific relational database products. The PC/IXF file format preserves this flexibility; for

PC Version of IXF File Format

example, the PC/IXF architecture supports both single-byte character string (SBCS) and double-byte character string (DBCS) data types. Not all implementations support all PC/IXF data types; however, even restricted implementations provide for the detection and disposition of unsupported data types during import.

In general, a PC/IXF file consists of an unbroken sequence of variable-length records. The file contains the following record types in the order shown:

- One header record of record type H
- One table record of record type T
- Multiple column descriptor records of record type C (one record for each column in the table)
- Multiple data records of record type D (each row in the table is represented by one or more D records).

A PC/IXF file may also contain application records of record type A, anywhere after the H record. These records are permitted in PC/IXF files to enable an application to include additional data, not defined by the PC/IXF format, in a PC/IXF file. A records are ignored by any program reading a PC/IXF file that does not have particular knowledge about the data format and content implied by the application identifier in the A record.

Every record in a PC/IXF file begins with a record length indicator. This is a 6-byte right justified character representation of an integer value specifying the length, in bytes, of the portion of the PC/IXF record that follows the record length indicator; that is, the total record size minus 6 bytes. Programs reading PC/IXF files should use these record lengths to locate the end of the current record and the beginning of the next record. H, T, and C records must be sufficiently large to include all of their defined fields, and, of course, their record length fields must agree with their actual lengths. However, if extra data (for example, a *new* field), is added to the end of one of these records, pre-existing programs reading PC/IXF files should ignore the extra data, and generate no more than a warning message. Programs writing PC/IXF files, however, should write H, T and C records that are the precise length needed to contain all of the defined fields.

PC/IXF file records are composed of fields which contain character data. The import and export utilities interpret this character data using the CPGID of the target database, with two exceptions:

- The IXFADATA field of A records.

The code page environment of character data contained in an IXFADATA field is established by the application which creates and processes a particular A record; that is, the environment varies by implementation.

- The IXFDCOLS field of D records.

The code page environment of character data contained in an IXFDCOLS field is a function of information contained in the C record which defines a particular column and its data.

PC Version of IXF File Format

Numeric fields in H, T, and C records, and in the prefix portion of D and A records should be right justified single-byte character representations of integer values, filled with leading zeros or blanks. A value of zero should be indicated with at least one (right justified) zero character, not blanks. Whenever one of these numeric fields is not used, for example IXFLENGL, where the length is implied by the data type, it should be filled with blanks. These numeric fields are:

IXFHRECL, IXFTRECL, IXFCRECL, IXFDRECL, IXFARECL,
IXFHHCNT, IXFHSBCP, IXFHDBCP, IXFTCCNT, IXFTNAML,
IXFLENGL, IXFCDRID, IXFCPOSN, IXFCNAML, IXFCTYPE,
IXFCSBCP, IXFCDBCP, IXFCNDIM, IXFCDSIZ, IXFDRID

Note: The database manager PC/IXF file format is not identical to the System/370 IXF format (see "Differences between Version 1 PC/IXF and Version 0 System/370 IXF" on page 470).

PC/IXF Record Types

There are five PC/IXF record types:

- header
- table
- column descriptor
- data
- application

Each PC/IXF record type is defined as a sequence of fields; these fields are required, and must appear in the order shown.

HEADER RECORD			
FIELD NAME	LENGTH	TYPE	COMMENTS
-----	-----	-----	-----
IXFHRECL	06-BYTE	CHARACTER	record length
IXFHRECT	01-BYTE	CHARACTER	record type = 'H'
IXFHID	03-BYTE	CHARACTER	IXF identifier
IXFHVERS	04-BYTE	CHARACTER	IXF version
IXFHPROD	12-BYTE	CHARACTER	product
IXFHDATE	08-BYTE	CHARACTER	date written
IXFHTIME	06-BYTE	CHARACTER	time written
IXFHHCNT	05-BYTE	CHARACTER	heading record count
IXFHSBCP	05-BYTE	CHARACTER	single byte code page
IXFHDBCP	05-BYTE	CHARACTER	double byte code page
IXHFIL1	02-BYTE	CHARACTER	reserved

The following fields are contained in the header record:

IXFHRECL The record length indicator. A 6-byte character representation of an integer value specifying the length, in bytes, of the portion of the PC/IXF record that follows the record length indicator; that is, the total

PC Version of IXF File Format

record size minus 6 bytes. The H record must be sufficiently long to include all of its defined fields.

IXFHRECT	The IXF record type, which is set to H for this record.
IXFHID	The file format identifier, which is set to IXF for this file.
IXFHVERS	The PC/IXF format level used when the file was created, which is set to '0001'.
IXFHPROD	A field that can be used by the program creating the file to identify itself. If this field is filled in, the first six bytes are used to identify the product creating the file, and the last six bytes are used to indicate the version or release of the creating product. The database manager uses this field to signal the existence of database manager-specific data.
IXFHDATE	The date on which the file was written, in the form <i>yyyymmdd</i> .
IXFHTIME	The time at which the file was written, in the form <i>hhmmss</i> . This field is optional and can be left blank.
IXFHHCNT	The number of H, T, and C records in this file that precede the first data record. A records are not included in this count.
IXFHSBCP	Single-byte code page field, containing a single-byte character representation of a SBCS CPGID or '00000'. The export utility sets this field equal to the SBCS CPGID of the exported database table. For example, if the table SBCS CPGID is 850, this field contains '00850'.
IXFHDBCP	Double-byte code page field, containing a single-byte character representation of a DBCS CPGID or '00000'. The export utility sets this field equal to the DBCS CPGID of the exported database table. For example, if the table DBCS CPGID is 301, this field contains '00301'.
IXFHFIL1	Spare field set to two blanks to match a reserved field in host IXF files.

PC Version of IXF File Format

TABLE RECORD			
FIELD NAME	LENGTH	TYPE	COMMENTS
-----	-----	-----	-----
IXFTRECL	06-BYTE	CHARACTER	record length
IXFTRECT	01-BYTE	CHARACTER	record type = 'T'
IXFTNAML	02-BYTE	CHARACTER	name length
IXFTNAME	18-BYTE	CHARACTER	name of data
IXFTQUAL	08-BYTE	CHARACTER	qualifier
IXFTSRC	12-BYTE	CHARACTER	data source
IXFTDATA	01-BYTE	CHARACTER	data convention = 'C'
IXFTFORM	01-BYTE	CHARACTER	data format = 'M'
IXFTMFRM	05-BYTE	CHARACTER	machine format='PC'
IXFTLOC	01-BYTE	CHARACTER	data location = 'I'
IXFTCCNT	05-BYTE	CHARACTER	'C' record count
IXFTFIL1	02-BYTE	CHARACTER	reserved
IXFTDESC	30-BYTE	CHARACTER	data description

The following fields are contained in the table record:

- IXFTRECL** The record length indicator. A 6-byte character representation of an integer value specifying the length, in bytes, of the portion of the PC/IXF record that follows the record length indicator; that is, the total record size minus 6 bytes. The T record must be sufficiently long to include all of its defined fields.
- IXFTRECT** The IXF record type, which is set to T for this record.
- IXFTNAML** The length, in bytes, of the table name in the IXFTNAME field.
- IXFTNAME** The name of the table. If each file has only one table, this is an informational field only. The database manager does not use this field when importing data. When writing a PC/IXF file, the database manager writes the DOS file name (and possibly path information) to this field.
- IXFTQUAL** Table name qualifier, which identifies the creator of a table in a relational system. This is an informational field only. If a program writing a file has no data to write to this field, the preferred fill value is blanks. Programs reading a file may print or display this field, or store it in an informational field, but no computations should depend on the content of this field.
- IXFTSRC** Used to indicate the original source of the data. This is an informational field only. If a program writing a file has no data to write to this field, the preferred fill value is blanks. Programs reading a file may print or display this field, or store it in an informational field, but no computations should depend on the content of this field.

PC Version of IXF File Format

IXFTDATA	Convention used to describe the data. This field must be set to C for import and export, indicating that individual column attributes are described in the following column descriptor (C) records, and that data follows PC/IXF conventions.
IXFTFORM	Convention used to store numeric data. This field must be set to M, indicating that numeric data in the data (D) records is stored in the machine (internal) format specified by the IXFTMFRM field.
IXFTMFRM	The format of any machine data in the PC/IXF file. The database manager will only read or write files if this field is set to <i>PCbbb</i> , where <i>b</i> represents a blank, and PC specifies that data in the PC/IXF file is in IBM PC machine format.
IXFTLOC	The location of the data. The database manager only supports a value of I, meaning the data is internal to this file.
IXFTCNT	The number of C records in this table. It is a right-justified character representation of an integer value.
IXFTFIL1	Spare field set to two blanks to match a reserved field in host IXF files.
IXFTDESC	Descriptive data about the table. This is an informational field only. If a program writing a file has no data to write to this field, the preferred fill value is blanks. Programs reading a file may print or display this field, or store it in an informational field, but no computations should depend on the content of this field. This field contains NOT NULL WITH DEFAULT if the column was not null with default, and the table name came from a workstation database.

PC Version of IXF File Format

COLUMN DESCRIPTOR RECORD			
FIELD NAME	LENGTH	TYPE	COMMENTS
-----	-----	-----	-----
IXFCRECL	06-BYTE	CHARACTER	record length
IXFCRECT	01-BYTE	CHARACTER	record type = 'C'
IXFCNAML	02-BYTE	CHARACTER	column name length
IXFCNAME	18-BYTE	CHARACTER	column name
IXFCNULL	01-BYTE	CHARACTER	column allows nulls
IXFCSLCT	01-BYTE	CHARACTER	column selected flag
IXFCKEY	01-BYTE	CHARACTER	key column flag
IXFCCLAS	01-BYTE	CHARACTER	data class
IXFCTYPE	03-BYTE	CHARACTER	data type
IXFCSBCP	05-BYTE	CHARACTER	single byte code page
IXFCDBC	05-BYTE	CHARACTER	double byte code page
IXFCLENG	05-BYTE	CHARACTER	column data length
IXFCDRID	03-BYTE	CHARACTER	'D' record identifier
IXFCPOSN	06-BYTE	CHARACTER	column position
IXFCDESC	30-BYTE	CHARACTER	column description
IXFCNDIM	02-BYTE	CHARACTER	number of dimensions
IXFCDSIZ	varying	CHARACTER	size of each dimension

The following fields are contained in column descriptor records:

- IXFCRECL** The record length indicator. A 6-byte character representation of an integer value specifying the length, in bytes, of the portion of the PC/IXF record that follows the record length indicator; that is, the total record size minus 6 bytes. The C record must be sufficiently long to include all of its defined fields.
- IXFCRECT** The IXF record type, which is set to C for this record.
- IXFCNAML** The length, in bytes, of the column name in the IXFCNAME field.
- IXFCNAME** The name of the column.
- IXFCNULL** Specifies if nulls are permitted in this column. Valid settings are Y or N.
- IXFCSLCT** An obsolete field whose intended purpose was to allow selection of a subset of columns in the data. Programs writing PC/IXF files should always store a Y in this field. Programs reading PC/IXF files should ignore the field.
- IXFCKEY** The key indicator. If the value of this field is Y, the column is a key column; if the value is N, the column is not a key column. The database manager does not use this field. It ignores the field when importing data, and sets it to N when generating an export file.
- IXFCCLAS** The class of data types to be used in the IXFCTYPE field. The database manager only supports relational types (R).

PC Version of IXF File Format

IXFCTYPE	<p>The data type for the column. For more information about data types, see “PC/IXF Data Types” on page 449.</p>
IXFCSBCP	<p>Contains a single-byte character representation of a SBCS CPGID. This field specifies the CPGID for single-byte character data, which occurs with the IXFDCOLS field of the D records for this column.</p> <p>The semantics of this field vary with the data type for the column (specified in the IXFCTYPE field).</p> <ul style="list-style-type: none">• For a character string column, this field should normally contain a nonzero value equal to that of the IXFHSSBCP field in the H record; however, other values are permitted. If this value is zero, the column is interpreted to contain bit string data.• For a numeric column, this field is not meaningful. It is set to zero by the export utility, and ignored by the import utility.• For a date or time column, this field is not meaningful. It is set to the value of the IXFHSSBCP field by the export utility, and ignored by the import utility.• For a graphic column, this field must be zero. <p>See also Table 12 on page 454.</p>
IXFDCBCP	<p>Contains a single-byte character representation of a DBCS CPGID. This field specifies the CPGID for double-byte character data, which occurs with the IXFDCOLS field of the D records for this column.</p> <p>The semantics of this field vary with the data type for the column (specified in the IXFCTYPE field).</p> <ul style="list-style-type: none">• For a character string column, this field should either be zero, or contain a value equal to that of the IXFHDBCPC field in the H record; however, other values are permitted. If the value in the IXFCSBCP field is zero, the value in this field must be zero.• For a numeric column, this field is not meaningful. It is set to zero by the export utility, and ignored by the import utility.• For a date or time column, this field is not meaningful. It is set to zero by the export utility, and ignored by the import utility.• For a graphic column, this field must have a value equal to the value of the IXFHDBCPC field. <p>See also Table 12 on page 454.</p>
IXFCLENG	<p>Provides information about the size of the column being described. For some data types, this field is unused, and should contain blanks. For other data types, this field contains the right-justified character representation of an integer specifying the column length. For yet other data types, this field is divided into two subfields: 3 bytes for precision, and 2 bytes for scale; both of these subfields are right-justified character representations of integers.</p>
IXFCDRID	<p>The D record identifier. This field contains the right-justified character representation of an integer value. Several D records can be used to contain each row of data in the PC/IXF file. This field specifies which</p>

PC Version of IXF File Format

D record (of the several D records contributing to a row of data) contains the data for the column. A value of one (for example, 001) indicates that the data for a column is in the first D record in a row of data. The first C record must have an IXFCDRID value of one. All subsequent C records must have an IXFCDRID value equal to the value in the preceding C record, or one higher.

IXFCPOSN The value in this field is used to locate the data for the column within one of the D records representing a row of table data. It is the starting position of the data for this column within the IXFCOLS field of the D record. If the column is nullable, IXFCPOSN points to the null indicator; otherwise, it points to the data itself. If a column contains varying length data, the data itself begins with the current length indicator. The IXFCPOSN value for the first byte in the IXFCOLS field of the D record is one (not zero). If a column is in a new D record, the value of IXFCPOSN should be one; otherwise, IXFCPOSN values should increase from column to column to such a degree that the data values do not overlap.

IXFCDESC Descriptive information about the column. This is an informational field only. If a program writing to a file has no data to write to this field, the preferred fill value is blanks. Programs reading a file may print or display this field, or store it in an informational field, but no computations should depend on the content of this field. If

- the column is not null with default
- the table resides in a workstation database
- the select statement on the export is of the form `select * from table`

the export utility will put `NOT NULL WITH DEFAULT` in this field, and when the import utility creates a new table with this file, it will create it as not null with default.

IXFCNDIM The number of dimensions in the column. Arrays are not supported in this version of PC/IXF. This field must therefore contain a character representation of a zero integer value.

IXFCDSIZ The size or range of each dimension. The length of this field is five bytes per dimension. Since arrays are not supported (that is, the number of dimensions must be zero), this field has zero length, and does not actually exist.

PC Version of IXF File Format

DATA RECORD			
FIELD NAME	LENGTH	TYPE	COMMENTS
-----	-----	-----	-----
IXFDRECL	06-BYTE	CHARACTER	record length
IXFDRECT	01-BYTE	CHARACTER	record type = 'D'
IXFDRID	03-BYTE	CHARACTER	'D' record identifier
IXFDFIL1	04-BYTE	CHARACTER	reserved
IXFDCOLS	varying	variable	columnar data

The following fields are contained in the data records:

- IXFDRECL** The record length indicator. A 6-byte character representation of an integer value specifying the length, in bytes, of the portion of the PC/IXF record that follows the record length indicator; that is, the total record size minus 6 bytes. Each D record must be sufficiently long to include all significant data for the current occurrence of the last data column stored in the record.
- IXFDRECT** The IXF record type, which is set to D for this record, indicating that it contains data values for the table.
- IXFDRID** The record identifier, which identifies a particular D record within the sequence of several D records contributing to a row of data. For the first D record in a row of data, this field has a value of one; for the second D record in a row of data, this field has a value of two, and so on. In each row of data, all the D record identifiers called out in the C records must actually exist.
- IXFDFIL1** Spare field set to four blanks to match reserved fields, and hold a place for a possible shift-out character, in host IXF files.
- IXFDCOLS** The area for columnar data. The data area of a data record (D record) is composed of one or more column entries. There is one column entry for each column descriptor record, which has the same D record identifier as the D record. In the D record, the starting position of the column entries is indicated by the IXFCPOSN value in the C records.
- The format of the column entry data depends on whether or not the column is nullable:
- If the column is nullable (the IXFCNULL field is set to Y), the column entry data includes a null indicator. If the column is not null, the indicator is followed by data type-specific information, including the actual database value. The null indicator is a two-byte value set to x'0000' for not null, and x'FFFF' for null.
 - If the column is not nullable, the column entry data includes only data type-specific information, including the actual database value.

PC Version of IXF File Format

For varying-length data types, the data type-specific information includes a current length indicator. The current length indicators are 2-byte integers in a form specified by the IXFTMFRM field.

The length of the data area of a D record may not exceed 32 771 bytes.

APPLICATION RECORD			
FIELD NAME	LENGTH	TYPE	COMMENTS
-----	-----	-----	-----
IXFARECL	06-BYTE	CHARACTER	record length
IXFARECT	01-BYTE	CHARACTER	record type = 'A'
IXFAPPID	12-BYTE	CHARACTER	application identifier
IXFADATA	varying	variable	application-specific data

The following fields are contained in application records:

- IXFARECL** The record length indicator. A 6-byte character representation of an integer value specifying the length, in bytes, of the portion of the PC/IXF record that follows the record length indicator; that is, the total record size minus 6 bytes. Each A record must be sufficiently long to include at least the entire IXFAPPID field.
- IXFARECT** The IXF record type, which is set to A for this record, indicating that this is an application record. These records are ignored by programs which do not have particular knowledge about the content and the format of the data implied by the application identifier.
- IXFAPPID** The application identifier, which identifies the application creating the A record. PC/IXF files created by the database manager may have A records with the first 6 characters of this field set to a constant identifying the database manager, and the last 6 characters identifying the release or version of the database manager or another application writing the A record.
- IXFADATA** This field contains application dependent supplemental data, whose form and content are known only to the program creating the A record, and to other applications which are likely to process the A record.

PC Version of IXF File Format

PC/IXF Data Types

Table 11 (Page 1 of 6). PC/IXF Data Types		
Name	IXFCTYPE Value	Description
BIGINT	492	An 8-byte integer in the form specified by IXFTMFRM. It represents a whole number between -9223 372 036 854 775 808 and 9223 372 036 854 775 807. IXFCSBCP and IXFCDBCP are not significant, and should be zero. IXFCLENG is not used, and should contain blanks.
BLOB, CLOB	404, 408	<p>A variable-length character string. The maximum length of the string is contained in the IXFCLENG field of the column descriptor record, and cannot exceed 32767 bytes. The string itself is preceded by a current length indicator, which is a 4-byte integer specifying the length of the string, in bytes. The string is in the code page indicated by IXFCSBCP.</p> <p>The following applies to BLOBs only: If IXFCSBCP is zero, the string is bit data, and should not be translated by any transformation program.</p> <p>The following applies to CLOBs only: If IXFCDBCP is nonzero, the string can also contain double-byte characters in the code page indicated by IXFCDBCP.</p>
BLOB_FILE, CLOB_FILE, DBCLOB_FILE	804, 808, 812	<p>A fixed-length field containing an SQLFILE structure with the <i>name_length</i> and the <i>name</i> fields filled in. The length of the structure is contained in the IXFCLENG field of the column descriptor record, and cannot exceed 255 bytes. The file name is in the code page indicated by IXFCSBCP. If IXFCDBCP is nonzero, the file name can also contain double-byte characters in the code page indicated by IXFCDBCP. If IXFCSBCP is zero, the file name is bit data and should not be translated by any transformation program.</p> <p>Since the length of the structure is stored in IXFCLENG, the actual length of the original LOB is lost. IXF files with columns of type BLOB_FILE, CLOB_FILE, or DBCLOB_FILE should not be used to recreate the LOB field, since the LOB will be created with a length of <i>sql_lobfile_len</i>.</p>

PC Version of IXF File Format

<i>Table 11 (Page 2 of 6). PC/IXF Data Types</i>		
Name	IXFCTYPE Value	Description
CHAR	452	A fixed-length character string. The string length is contained in the IXFLEN field of the column descriptor record, and cannot exceed 254 bytes. The string is in the code page indicated by IXFCSBCP. If IXFCDBCP is nonzero, the string can also contain double-byte characters in the code page indicated by IXFCDBCP. If IXFCSBCP is zero, the string is bit data and should not be translated by any transformation program.
DATE	384	A point in time in accordance with the Gregorian calendar. Each date is a 10-byte character string in International Standards Organization (ISO) format: <i>yyyy-mm-dd</i> . The range of the year part is 0001 to 9999. The range of the month part is 01 to 12. The range of the day part is 01 to <i>n</i> , where <i>n</i> depends on the month, using the usual rules for days of the month and leap year. Leading zeros cannot be omitted from any part. IXFLEN is not used, and should contain blanks. Valid characters within DATE are invariant in all PC ASCII code pages; therefore, IXFCSBCP and IXFCDBCP are not significant, and should be zero.
DBCLOB	412	A variable-length string of double-byte characters. The IXFLEN field in the column descriptor record specifies the maximum number of double-byte characters in the string, and cannot exceed 16383. The string itself is preceded by a current length indicator, which is a 4-byte integer specifying the length of the string in double-byte characters (that is, the value of this integer is one half the length of the string, in bytes). The string is in the DBCS code page, as specified by IXFCDBCP in the C record. Since the string consists of double-byte character data only, IXFCSBCP should be zero. There are no surrounding shift-in or shift-out characters.

PC Version of IXF File Format

Table 11 (Page 3 of 6). PC/IXF Data Types

Name	IXFCTYPE Value	Description
DECIMAL	484	A packed decimal number with precision P (as specified by the first three bytes of IXFLENG in the column descriptor record) and scale S (as specified by the last two bytes of IXFLENG). The length, in bytes, of a packed decimal number is $(P+2)/2$. The precision must be an odd number between 1 and 31, inclusive. The packed decimal number is in the internal format specified by IXFTMFRM, where packed decimal for the PC is defined to be the same as packed decimal for the System/370. IXFCSBCP and IXFCDBCP are not significant, and should be zero.
FLOATING POINT	480	Either a long (8-byte) or short (4-byte) floating point number, depending on whether IXFLENG is set to eight or to four. The data is in the internal machine form, as specified by IXFTMFRM. IXFCSBCP and IXFCDBCP are not significant, and should be zero. Four-byte floating point is not supported by the database manager.
GRAPHIC	468	A fixed-length string of double-byte characters. The IXFLENG field in the column descriptor record specifies the number of double-byte characters in the string, and cannot exceed 127. The actual length of the string is twice the value of the IXFLENG field, in bytes. The string is in the DBCS code page, as specified by IXFCDBCP in the C record. Since the string consists of double-byte character data only, IXFCSBCP should be zero. There are no surrounding shift-in or shift-out characters.
INTEGER	496	A 4-byte integer in the form specified by IXFTMFRM. It represents a whole number between -2 147 483 648 and +2 147 483 647. IXFCSBCP and IXFCDBCP are not significant, and should be zero. IXFLENG is not used, and should contain blanks.

PC Version of IXF File Format

Table 11 (Page 4 of 6). PC/IXF Data Types

Name	IXFCTYPE Value	Description
LONGVARCHAR	456	A variable-length character string. The maximum length of the string is contained in the IXFCLENG field of the column descriptor record, and cannot exceed 32 767 bytes. The string itself is preceded by a current length indicator, which is a 2-byte integer specifying the length of the string, in bytes. The string is in the code page indicated by IXFCSBCP. If IXFCDBCP is nonzero, the string can also contain double-byte characters in the code page indicated by IXFCDBCP. If IXFCSBCP is zero, the string is bit data and should not be translated by any transformation program.
LONG VARGRAPHIC	472	A variable-length string of double-byte characters. The IXFCLENG field in the column descriptor record specifies the maximum number of double-byte characters for the string, and cannot exceed 16 383. The string itself is preceded by a current length indicator, which is a 2-byte integer specifying the length of the string in double-byte characters (that is, the value of this integer is one half the length of the string, in bytes). The string is in the DBCS code page, as specified by IXFCDBCP in the C record. Since the string consists of double-byte character data only, IXFCSBCP should be zero. There are no surrounding shift-in or shift-out characters.
SMALLINT	500	A 2-byte integer in the form specified by IXFTMFRM. It represents a whole number between -32 768 and +32 767. IXFCSBCP and IXFCDBCP are not significant, and should be zero. IXFCLENG is not used, and should contain blanks.

PC Version of IXF File Format

Table 11 (Page 5 of 6). PC/IXF Data Types

Name	IXFCTYPE Value	Description
TIME	388	A point in time in accordance with the 24-hour clock. Each time is an 8-byte character string in ISO format: <i>hh.mm.ss</i> . The range of the hour part is 00 to 24, and the range of the other parts is 00 to 59. If the hour is 24, the other parts are 00. The smallest time is 00.00.00, and the largest is 24.00.00. Leading zeros cannot be omitted from any part. IXFCLENG is not used, and should contain blanks. Valid characters within TIME are invariant in all PC ASCII code pages; therefore, IXFCSBCP and IXFCDBCP are not significant, and should be zero.
TIMESTAMP	392	The date and time with microsecond precision. Each time stamp is a character string of the form <i>yyyy-mm-dd-hh.mm.ss.nnnnnn</i> (year month day hour minutes seconds microseconds). IXFCLENG is not used, and should contain blanks. Valid characters within TIMESTAMP are invariant in all PC ASCII code pages; therefore, IXFCSBCP and IXFCDBCP are not significant, and should be zero.
VARCHAR	448	A variable-length character string. The maximum length of the string, in bytes, is contained in the IXFCLENG field of the column descriptor record, and cannot exceed 254 bytes. The string itself is preceded by a current length indicator, which is a two-byte integer specifying the length of the string, in bytes. The string is in the code page indicated by IXFCSBCP. If IXFCDBCP is nonzero, the string can also contain double-byte characters in the code page indicated by IXFCDBCP. If IXFCSBCP is zero, the string is bit data and should not be translated by any transformation program.

PC Version of IXF File Format

Table 11 (Page 6 of 6). PC/IXF Data Types

Name	IXFCTYPE Value	Description
VARGRAPHIC	464	A variable-length string of double-byte characters. The IXFLEN field in the column descriptor record specifies the maximum number of double-byte characters in the string, and cannot exceed 127. The string itself is preceded by a current length indicator, which is a 2-byte integer specifying the length of the string in double-byte characters (that is, the value of this integer is one half the length of the string, in bytes). The string is in the DBCS code page, as specified by IXFCDBCP in the C record. Since the string consists of double-byte character data only, IXFCSBCP should be zero. There are no surrounding shift-in or shift-out characters.

Not all combinations of IXFCSBCP and IXFCDBCP values for PC/IXF character or graphic columns are valid. A PC/IXF character or graphic column with an invalid (IXFCSBCP,IXFCDBCP) combination is an invalid data type.

Table 12. Valid PC/IXF Data Types

PC/IXF Data Type	Valid (IXFCSBCP,IXFCDBCP) Pairs	Invalid (IXFCSBCP,IXFCDBCP) Pairs
CHAR, VARCHAR, or LONG VARCHAR	(0,0), (x,0), or (x,y)	(0,y)
BLOB	(0,0)	(x,0), (0,y), or (x,y)
CLOB	(x,0), (x,y)	(0,0), (0,y)
GRAPHIC, VARGRAPHIC, LONG VARGRAPHIC, or DBCLOB	(0,y)	(0,0), (x,0), or (x,y)
Note: x and y are not 0.		

PC/IXF Data Type Descriptions

<i>Table 13 (Page 1 of 5). Acceptable Data Type Forms for the PC/IXF File Format</i>		
Data Type	Form in Files Created by the Export Utility	Form Acceptable to the Import Utility
BIGINT	A BIGINT column, identical to the database column, is created.	A column in any numeric type (SMALLINT, INTEGER, BIGINT, DECIMAL, or FLOAT) is accepted. Individual values are rejected if they are not in the range -9 223 372 036 854 775 808 to 9 223 372 036 854 775 807.
BLOB	A PC/IXF BLOB column is created. The maximum length of the database column, the SBCS CPGID value, and the DBCS CPGID value are copied to the column descriptor record.	A PC/IXF CHAR, VARCHAR, LONG VARCHAR, BLOB, or BLOB_FILE column is acceptable if: <ul style="list-style-type: none"> • The database column is marked FOR BIT DATA • The PC/IXF column single-byte code page value equals the SBCS CPGID of the database column, and the PC/IXF column double-byte code page value equals zero, or the DBCS CPGID of the database column. A PC/IXF GRAPHIC, VARGRAPHIC, or LONG VARGRAPHIC BLOB column is also acceptable. If the PC/IXF column is of fixed length, its length must be compatible with the maximum length of the database column. See also the "FORCEIN Option" on page 463.
CHAR	A PC/IXF CHAR column is created. The database column length, the SBCS CPGID value, and the DBCS CPGID value are copied to the PC/IXF column descriptor record.	A PC/IXF CHAR, VARCHAR, or LONG VARCHAR column is acceptable if: <ul style="list-style-type: none"> • The database column is marked FOR BIT DATA • The PC/IXF column single-byte code page value equals the SBCS CPGID of the database column, and the PC/IXF column double-byte code page value equals zero, or the DBCS CPGID of the database column. A PC/IXF GRAPHIC, VARGRAPHIC, or LONG VARGRAPHIC column is also acceptable if the database column is marked FOR BIT DATA. In any case, if the PC/IXF column is of fixed length, its length must be compatible with the length of the database column. The data is padded on the right with single-byte spaces (x'20'), if necessary. See also the "FORCEIN Option" on page 463.

PC Version of IXF File Format

Table 13 (Page 2 of 5). Acceptable Data Type Forms for the PC/IXF File Format

Data Type	Form in Files Created by the Export Utility	Form Acceptable to the Import Utility
CLOB	A PC/IXF CLOB column is created. The maximum length of the database column, the SBCS CPGID value, and the DBCS CPGID value are copied to the column descriptor record.	A PC/IXF CHAR, VARCHAR, LONG VARCHAR, CLOB, or CLOB_FILE column is acceptable if the PC/IXF column single-byte code page value equals the SBCS CPGID of the database column, and the PC/IXF column double-byte code page value equals zero, or the DBCS CPGID of the database column. If the PC/IXF column is of fixed length, its length must be compatible with the maximum length of the database column. See also the "FORCEIN Option" on page 463.
DATE	A DATE column, identical to the database column, is created.	A PC/IXF column of type DATE is the usual input. The import utility also attempts to accept columns in any of the character types, except those with incompatible lengths. The character column in the PC/IXF file must contain dates in a format consistent with the country code of the target database.
DBCLOB	A PC/IXF DBCLOB column is created. The maximum length of the database column, the SBCS CPGID value, and the DBCS CPGID value are copied to the column descriptor record.	A PC/IXF GRAPHIC, VARGRAPHIC, LONG VARGRAPHIC, DBCLOB, or DBCLOB_FILE column is acceptable if the PC/IXF column double-byte code page value equals that of the database column. If the PC/IXF column is of fixed length, its length must be compatible with the maximum length of the database column. See also the "FORCEIN Option" on page 463.
DECIMAL	A DECIMAL column, identical to the database column, is created. The precision and scale of the column is stored in the column descriptor record.	A column in any numeric type (SMALLINT, INTEGER, BIGINT, DECIMAL, or FLOAT) is accepted. Individual values are rejected if they are not in the range of the DECIMAL column into which they are being imported.
FLOAT	A FLOAT column, identical to the database column, is created.	A column in any numeric type (SMALLINT, INTEGER, BIGINT, DECIMAL, or FLOAT) is accepted. All values are within range.

PC Version of IXF File Format

Table 13 (Page 3 of 5). Acceptable Data Type Forms for the PC/IXF File Format

Data Type	Form in Files Created by the Export Utility	Form Acceptable to the Import Utility
GRAPHIC (DBCS only)	A PC/IXF GRAPHIC column is created. The database column length, the SBCS CPGID value, and the DBCS CPGID value are copied to the column descriptor record.	A PC/IXF GRAPHIC, VARGRAPHIC, or LONG VARGRAPHIC column is acceptable if the PC/IXF column double-byte code page value equals that of the database column. If the PC/IXF column is of fixed length, its length must be compatible with the database column length. The data is padded on the right with double-byte spaces (x'8140'), if necessary. See also the "FORCEIN Option" on page 463.
INTEGER	An INTEGER column, identical to the database column, is created.	A column in any numeric type (SMALLINT, INTEGER, BIGINT, DECIMAL, or FLOAT) is accepted. Individual values are rejected if they are not in the range -2 147 483 648 to 2 147 483 647.
LONG VARCHAR	A PC/IXF LONG VARCHAR column is created. The maximum length of the database column, the SBCS CPGID value, and the DBCS CPGID value are copied to the column descriptor record.	<p>A PC/IXF CHAR, VARCHAR, or LONG VARCHAR column is acceptable if:</p> <ul style="list-style-type: none"> • The database column is marked FOR BIT DATA • The PC/IXF column single-byte code page value equals the SBCS CPGID of the database column, and the PC/IXF column double-byte code page value equals zero, or the DBCS CPGID of the database column. <p>A PC/IXF GRAPHIC, VARGRAPHIC, or LONG VARGRAPHIC column is also acceptable if the database column is marked FOR BIT DATA. In any case, if the PC/IXF column is of fixed length, its length must be compatible with the maximum length of the database column. See also the "FORCEIN Option" on page 463.</p>
LONG VARGRAPHIC (DBCS only)	A PC/IXF LONG VARGRAPHIC column is created. The maximum length of the database column, the SBCS CPGID value, and the DBCS CPGID value are copied to the column descriptor record.	A PC/IXF GRAPHIC, VARGRAPHIC, or LONG VARGRAPHIC column is acceptable if the PC/IXF column double-byte code page value equals that of the database column. If the PC/IXF column is of fixed length, its length must be compatible with the maximum length of the database column. See also the "FORCEIN Option" on page 463.

PC Version of IXF File Format

Table 13 (Page 4 of 5). Acceptable Data Type Forms for the PC/IXF File Format

Data Type	Form in Files Created by the Export Utility	Form Acceptable to the Import Utility
SMALLINT	A SMALLINT column, identical to the database column, is created.	A column in any numeric type (SMALLINT, INTEGER, BIGINT, DECIMAL, or FLOAT) is accepted. Individual values are rejected if they are not in the range -32 768 to 32 767.
TIME	A TIME column, identical to the database column, is created.	A PC/IXF column of type TIME is the usual input. The import utility also attempts to accept columns in any of the character types, except those with incompatible lengths. The character column in the PC/IXF file must contain time data in a format consistent with the country code of the target database.
TIMESTAMP	A TIMESTAMP column, identical to the database column, is created.	A PC/IXF column of type TIMESTAMP is the usual input. The import utility also attempts to accept columns in any of the character types, except those with incompatible lengths. The character column in the PC/IXF file must contain data in the input format for time stamps.
VARCHAR	If the maximum length of the database column is ≤ 254 , a PC/IXF VARCHAR column is created. If the maximum length of the database column is > 254 , a PC/IXF LONG VARCHAR column is created. The maximum length of the database column, the SBCS CPGID value, and the DBCS CPGID value are copied to the column descriptor record.	<p>A PC/IXF CHAR, VARCHAR, or LONG VARCHAR column is acceptable if:</p> <ul style="list-style-type: none"> • The database column is marked FOR BIT DATA • The PC/IXF column single-byte code page value equals the SBCS CPGID of the database column, and the PC/IXF column double-byte code page value equals zero, or the DBCS CPGID of the database column. <p>A PC/IXF GRAPHIC, VARGRAPHIC, or LONG VARGRAPHIC column is also acceptable if the database column is marked FOR BIT DATA. In any case, if the PC/IXF column is of fixed length, its length must be compatible with the maximum length of the database column. See also the "FORCEIN Option" on page 463.</p>

PC Version of IXF File Format

Table 13 (Page 5 of 5). Acceptable Data Type Forms for the PC/IXF File Format

Data Type	Form in Files Created by the Export Utility	Form Acceptable to the Import Utility
VARGRAPHIC (DBCS only)	<p>If the maximum length of the database column is <= 127, a PC/IXF VARGRAPHIC column is created.</p> <p>If the maximum length of the database column is > 127, a PC/IXF LONG VARGRAPHIC column is created.</p> <p>The maximum length of the database column, the SBCS CPGID value, and the DBCS CPGID value are copied to the column descriptor record.</p>	<p>A PC/IXF GRAPHIC, VARGRAPHIC, or LONG VARGRAPHIC column is acceptable if the PC/IXF column double-byte code page value equals that of the database column. If the PC/IXF column is of fixed length, its length must be compatible with the maximum length of the database column. See also the "FORCEIN Option" on page 463.</p>

General Rules Governing PC/IXF File Import into Databases

The database manager import utility applies the following general rules when importing a PC/IXF file in either an SBCS or a DBCS environment:

- The import utility accepts PC/IXF format files only (IXFHID = 'IXF'). IXF files of other formats cannot be imported.
- The import utility rejects a PC/IXF file with more than 1024 columns.
- The value of IXFHSBCP in the PC/IXF H record must equal the SBCS CPGID, or there must be a conversion table between the IXFHSBCP/IXFHDBCP and the SBCS/DBCS CPGID of the target database. The value of IXFHDBCP must equal either '00000', or the DBCS CPGID of the target database. If either of these conditions is not satisfied, the import utility rejects the PC/IXF file, unless the "FORCEIN Option" on page 463 is specified.
- Invalid Data Types — New Table

Import of a PC/IXF file into a *new* table is specified by the CREATE or the REPLACE_CREATE keywords in the IMPORT command. If a PC/IXF column of an invalid data type (valid data types are defined in "PC/IXF Data Types" on page 449) is selected for import into a new table, the import utility terminates. The entire PC/IXF file is rejected, no table is created, and no data is imported.
- Invalid Data Types — Existing Table

PC Version of IXF File Format

Import of a PC/IXF file into an *existing* table is specified by the INSERT, the INSERT_UPDATE, or the REPLACE_CREATE keywords in the IMPORT command. If a PC/IXF column of an invalid data type is selected for import into an existing table, one of two actions is possible:

- If the target table column is nullable, all values for the invalid PC/IXF column are ignored, and the table column values are set to NULL
 - If the target table column is not nullable, the import utility terminates. The entire PC/IXF file is rejected, and no data is imported. The existing table remains unaltered.
- When importing into a new table, nullable PC/IXF columns generate nullable database columns, and not nullable PC/IXF columns generate not nullable database columns.
 - A not nullable PC/IXF column can be imported into a nullable database column.
 - A nullable PC/IXF column can be imported into a not nullable database column. If a NULL value is encountered in the PC/IXF column, the import utility rejects the values of all columns in the PC/IXF row that contains the NULL value (the entire row is rejected), and processing continues with the next PC/IXF row. That is, no data is imported from a PC/IXF row that contains a NULL value if a target table column (for the NULL) is not nullable.
 - Incompatible Columns — New Table

If, during import to a *new* database table, a PC/IXF column is selected that is incompatible with the target database column, the import utility terminates. The entire PC/IXF file is rejected, no table is created, and no data is imported.

Note: The IMPORT “FORCEIN Option” on page 463 extends the scope of compatible columns.

- Incompatible Columns — Existing Table

If, during import to an *existing* database table, a PC/IXF column is selected that is incompatible with the target database column, one of two actions is possible:

- If the target table column is nullable, all values for the PC/IXF column are ignored, and the table column values are set to NULL
- If the target table column is not nullable, the import utility terminates. The entire PC/IXF file is rejected, and no data is imported. The existing table remains unaltered.

Note: The IMPORT “FORCEIN Option” on page 463 extends the scope of compatible columns.

- Invalid Values

If, during import, a PC/IXF column value is encountered that is not valid for the target database column, the import utility rejects the values of all columns in the PC/IXF row that contains the invalid value (the entire row is rejected), and processing continues with the next PC/IXF row.

Data Type-Specific Rules Governing PC/IXF File Import into Databases

- A valid PC/IXF numeric column can be imported into any compatible numeric database column. PC/IXF columns containing 4-byte floating point data are not imported, because this is an invalid data type.
- Database date/time columns can accept values from matching PC/IXF date/time columns (DATE, TIME, and TIMESTAMP), as well as from PC/IXF character columns (CHAR, VARCHAR, and LONG VARCHAR), subject to column length and value compatibility restrictions.
- A valid PC/IXF character column (CHAR, VARCHAR, or LONG VARCHAR) can always be imported into an *existing* database character column marked FOR BIT DATA; otherwise:
 - IXFCSBCP and the SBCS CPGID must agree
 - There must be a conversion table for the IXFCSBCP/IXFCDBCP and the SBCS/DBCS
 - One set must be all zeros (FOR BIT DATA).

If IXFCSBCP is not zero, the value of IXFCDBCP must equal either zero or the DBCS CPGID of the target database column.

If either of these conditions is not satisfied, the PC/IXF and database columns are incompatible.

When importing a valid PC/IXF character column into a *new* database table, the value of IXFCSBCP must equal either zero or the SBCS CPGID of the database, or there must be a conversion table. If IXFCSBCP is zero, IXFCDBCP must also be zero (otherwise the PC/IXF column is an invalid data type); IMPORT creates a character column marked FOR BIT DATA in the new table. If IXFCSBCP is not zero, and equals the SBCS CPGID of the database, the value of IXFCDBCP must equal either zero or the DBCS CPGID of the database; in this case, the utility creates a character column in the new table with SBCS and DBCS CPGID values equal to those of the database. If these conditions are not satisfied, the PC/IXF and database columns are incompatible.

The “FORCEIN Option” on page 463 can be used to override code page equality checks. However, a PC/IXF character column with IXFCSBCP equal to zero and IXFCDBCP not equal to zero is an invalid data type, and cannot be imported, even if FORCEIN is specified.

- A valid PC/IXF graphic column (GRAPHIC, VARGRAPHIC, or LONG VARGRAPHIC) can always be imported into an *existing* database character column marked FOR BIT DATA, but is incompatible with all other database columns. The “FORCEIN Option” on page 463 can be used to relax this restriction. However, a PC/IXF graphic column with IXFCSBCP not equal to zero, or IXFCDBCP equal to zero, is an invalid data type, and cannot be imported, even if FORCEIN is specified.

When importing a valid PC/IXF graphic column into a database graphic column, the value of IXFCDBCP must equal the DBCS CPGID of the target database column (that is, the double-byte code pages of the two columns must agree).

PC Version of IXF File Format

- If, during import of a PC/IXF file into an existing database table, a fixed-length string column (CHAR or GRAPHIC) is selected whose length is greater than the maximum length of the target column, the columns are incompatible.
- If, during import of a PC/IXF file into an existing database table, a variable-length string column (VARCHAR, LONG VARCHAR, VARGRAPHIC, or LONG VARGRAPHIC) is selected whose length is greater than the maximum length of the target column, the columns *are* compatible. Individual values are processed according to the compatibility rules governing the database manager INSERT statement, and PC/IXF values which are too long for the target database column are invalid.
- PC/IXF values imported into a fixed-length database *character* column (that is, a CHAR column) are padded on the right with single-byte spaces (0x20), if necessary, to obtain values whose length equals that of the database column. PC/IXF values imported into a fixed-length database *graphic* column (that is, a GRAPHIC column) are padded on the right with double-byte spaces (0x8140), if necessary, to obtain values whose length equals that of the database column.
- Since PC/IXF VARCHAR columns have a maximum length of 254 bytes, a database VARCHAR column of maximum length n , with $254 < n < 4001$, must be exported into a PC/IXF LONG VARCHAR column of maximum length n .
- Although PC/IXF LONG VARCHAR columns have a maximum length of 32 767 bytes, and database LONG VARCHAR columns have a maximum length restriction of 32 700 bytes, PC/IXF LONG VARCHAR columns of length greater than 32 700 bytes (but less than 32 768 bytes) are still valid, and can be imported into database LONG VARCHAR columns, but data may be lost.
- Since PC/IXF VARGRAPHIC columns have a maximum length of 127 bytes, a database VARGRAPHIC column of maximum length n , with $127 < n < 2001$, must be exported into a PC/IXF LONG VARGRAPHIC column of maximum length n .
- Although PC/IXF LONG VARGRAPHIC columns have a maximum length of 16 383 bytes, and database LONG VARGRAPHIC columns have a maximum length restriction of 16 350, PC/IXF LONG VARGRAPHIC columns of length greater than 16 350 bytes (but less than 16 384 bytes) are still valid, and can be imported into database LONG VARGRAPHIC columns, but data may be lost.

Table 14 summarizes PC/IXF file import into new or existing database tables without the FORCEIN option.

Table 14 (Page 1 of 2). Summary of PC/IXF File Import without FORCEIN Option												
PC/IXF COLUMN DATA TYPE	DATABASE COLUMN DATA TYPE											
	NUMERIC					CHARACTER			GRAPH	DATETIME		
	SMALL INT	INT	BIGINT	DEC	FLT	(0,0)	(SBCS, 0) ^d	(SBCS, DBCS) ^b	b	DATE	TIME	TIME STAMP
Numeric												
-SMALLINT	N											
	E	E	E	E ^a	E							
-INTEGER		N										
	E ^a	E	E	E ^a	E							

PC Version of IXF File Format

Table 14 (Page 2 of 2). Summary of PC/IXF File Import without FORCEIN Option

PC/IXF COLUMN DATA TYPE	DATABASE COLUMN DATA TYPE											
	NUMERIC					CHARACTER			GRAPH	DATETIME		
	SMALL INT	INT	BIGINT	DEC	FLT	(0,0)	(SBCS, 0) ^d	(SBCS, DBCS) ^b	b	DATE	TIME	TIME STAMP
-BIGINT			N									
	E ^a	E ^a	E	E ^a	E							
-DECIMAL				N								
	E ^a	E ^a	E ^a	E ^a	E							
-FLOAT					N							
	E ^a	E ^a	E ^a	E ^a	E							
Character												
-(0,0)						N						
						E				E ^c	E ^c	E ^c
-(SBCS,0)							N	N				
						E	E	E		E ^c	E ^c	E ^c
-(SBCS, DBCS)								N		E ^c	E ^c	E ^c
						E		E				
Graphic												
									N			
						E			E			
Datetime												
-DATE										N		
										E		
-TIME											N	
											E	
-TIME STAMP												N
												E
Notes:												
1. The table is a matrix of all valid PC/IXF and database manager data types. If a PC/IXF column can be imported into a database column, a letter is displayed in the matrix cell at the intersection of the PC/IXF data type matrix row and the database manager data type matrix column. An 'N' indicates that the utility is creating a new database table (a database column of the indicated data type is created). An 'E' indicates that the utility is importing data to an existing database table (a database column of the indicated data type is a valid target).												
2. Character string data types are distinguished by code page attributes. These attributes are shown as an ordered pair (SBCS,DBCS), where:												
• SBCS is either zero or denotes a nonzero value of the single-byte code page attribute of the character data type												
• DBCS is either zero or denotes a nonzero value of the double-byte code page attribute of the character data type.												
3. If the table indicates that a PC/IXF character column can be imported into a database character column, the values of their respective code page attribute pairs satisfy the rules governing code page equality.												
^a Individual values are rejected if they are out of range for the target numeric data type.												
^b Data type is available only in DBCS environments.												
^c Individual values are rejected if they are not valid date or time values.												
^d Data type is not available in DBCS environments.												

FORCEIN Option

The FORCEIN option permits import of a PC/IXF file despite code page differences between data in the PC/IXF file and the target database. It offers additional flexibility in the definition of compatible columns.

PC Version of IXF File Format

FORCEIN General Semantics

The following general semantics apply when using the FORCEIN option in either an SBCS or a DBCS environment:

- The FORCEIN option should be used with caution. It is usually advisable to attempt an import without this option enabled. However, because of the generic nature of the PC/IXF data interchange architecture, some PC/IXF files may contain data types or values that cannot be imported without intervention.
- Import with FORCEIN to a *new* table may yield a different result than import to an existing table. An existing table has predefined target data types for each PC/IXF data type.
- When LOB data is exported with the LOBSINFILE option, and the files move to another client with a different code page, then, unlike other data, the CLOBS and DBCLOBS in the separate files are not converted to the client code page when imported or loaded into a database.

FORCEIN Code Page Semantics

The following code page semantics apply when using the FORCEIN option in either an SBCS or a DBCS environment:

- The FORCEIN option disables all import utility code page comparisons.

This rule applies to code page comparisons at the column level and at the file level as well, when importing to a new or an existing database table. At the column (for example, data type) level, this rule applies only to the following database manager and PC/IXF data types: character (CHAR, VARCHAR, and LONG VARCHAR), and graphic (GRAPHIC, VARGRAPHIC, and LONG VARGRAPHIC). The restriction follows from the fact that code page attributes of other data types are not relevant to the interpretation of data type values.
- The FORCEIN option does not disable inspection of code page attributes to determine data types.

For example, the database manager allows a CHAR column to be declared with the FOR BIT DATA attribute. Such a declaration sets both the SBCS CPGID and the DBCS CPGID of the column to zero; it is the zero value of these CPGIDs that identifies the column values as bit strings (rather than character strings).
- The FORCEIN option does not imply code page translation.

Values of data types that are sensitive to the FORCEIN option are copied "as is". No code point mappings are employed to account for a change of code page environments. Padding of the imported value with spaces may be necessary in the case of fixed length target columns.
- When data is imported to an *existing* table using the FORCEIN option:
 - The code page value of the target database table and columns always prevails.
 - The code page value of the PC/IXF file and columns is ignored.

PC Version of IXF File Format

This rule applies whether or not the FORCEIN option is used. The database manager does not permit changes to a database or a column code page value once a database is created.

- When importing to a *new* table using the FORCEIN option:
 - The code page value of the target database prevails.
 - PC/IXF character columns with IXFCSBCP = IXFCDBCP = 0 generate table columns marked FOR BIT DATA.
 - All other PC/IXF character columns generate table character columns with SBCS and DBCS CPGID values equal to those of the database.
 - PC/IXF graphic columns generate table graphic columns with an SBCS CPGID of "undefined", and a DBCS CPGID equal to that of the database (DBCS environment only).

FORCEIN Example

Consider a PC/IXF CHAR column with IXFCSBCP = '00897' and IXFCDBCP = '00301'. This column is to be imported into a database CHAR column whose SBCS CPGID = '00850' and DBCS CPGID = '00000'. Without FORCEIN, the utility terminates, and no data is imported, or the PC/IXF column values are ignored, and the database column contains NULLs (if the database column is nullable). With FORCEIN, the utility proceeds, ignoring code page incompatibilities. If there are no other data type incompatibilities (such as length, for example), the values of the PC/IXF column are imported "as is", and become available for interpretation under the database column code page environment.

The following table shows:

- The code page attributes of a column created in a *new* database table when a PC/IXF file data type with specified code page attributes is imported
- That the import utility rejects PC/IXF data types if they invalid or incompatible.

Table 15 (Page 1 of 2). Summary of Import Utility Code Page Semantics (New Table). This table assumes there is no conversion table between a and x. If there were, items 3 and 4 would work successfully without the FORCEIN option.

CODE PAGE ATTRIBUTES of PC/IXF DATA TYPE	CODE PAGE ATTRIBUTES OF DATABASE TABLE COLUMN	
	Without FORCEIN	With FORCEIN
SBCS		
(0,0)	(0,0)	(0,0)
(a,0)	(a,0)	(a,0)
(x,0)	reject	(a,0)
(x,y)	reject	(a,0)
(a,y)	reject	(a,0)
(0,y)	reject	(0,0)
DBCS		

PC Version of IXF File Format

Table 15 (Page 2 of 2). Summary of Import Utility Code Page Semantics (New Table). This table assumes there is no conversion table between a and x. If there were, items 3 and 4 would work successfully without the FORCEIN option.

CODE PAGE ATTRIBUTES of PC/IXF DATA TYPE	CODE PAGE ATTRIBUTES OF DATABASE TABLE COLUMN	
	Without FORCEIN	With FORCEIN
(0,0)	(0,0)	(0,0)
(a,0)	(a,b)	(a,b)
(x,0)	reject	(a,b)
(a,b)	(a,b)	(a,b)
(x,y)	reject	(a,b)
(a,y)	reject	(a,b)
(x,b)	reject	(a,b)
(0,b)	(-,b)	(-,b)
(0,y)	reject	(-,b)

Notes:

- Code page attributes of a PC/IXF data type are shown as an ordered pair, where x represents a nonzero single-byte code page value, and y represents a nonzero double-byte code page value. A '-' represents an undefined code page value.
- The use of different letters in various code page attribute pairs is deliberate. Different letters imply different values. For example, if a PC/IXF data type is shown as (x,y), and the database column as (a,y), x does not equal a, but the PC/IXF file and the database have the same double-byte code page value y.
- Only character and graphic data types are affected by the FORCEIN code page semantics.
- It is assumed that the database containing the new table has code page attributes of (a,0); therefore, all character columns in the new table must have code page attributes of either (0,0) or (a,0).

In a DBCS environment, it is assumed that the database containing the new table has code page attributes of (a,b); therefore, all graphic columns in the new table must have code page attributes of (-,b), and all character columns must have code page attributes of (a,b). The SBCS CPGID is shown as '-', because it is undefined for graphic data types.

- The data type of the result is determined by the rules described in "FORCEIN Data Type Semantics" on page 468.
- The reject result is a reflection of the rules for invalid or incompatible data types (see "General Rules Governing PC/IXF File Import into Databases" on page 459).

PC Version of IXF File Format

The following table shows:

- That the import utility accepts PC/IXF data types with various code page attributes into an *existing* table column (the *target* column) having the specified code page attributes
- That the import utility does not permit a PC/IXF data type with certain code page attributes to be imported into an *existing* table column having the code page attributes shown. The utility rejects PC/IXF data types if they are invalid or incompatible.

Table 16 (Page 1 of 2). Summary of Import Utility Code Page Semantics (Existing Table). This table assumes there is no conversion table between a and x.

CODE PAGE ATTRIBUTES OF PC/IXF DATA TYPE	CODE PAGE ATTRIBUTES OF TARGET DATABASE COLUMN	RESULTS OF IMPORT	
		Without FORCEIN	With FORCEIN
SBCS			
(0,0)	(0,0)	accept	accept
(a,0)	(0,0)	accept	accept
(x,0)	(0,0)	accept	accept
(x,y)	(0,0)	accept	accept
(a,y)	(0,0)	accept	accept
(0,y)	(0,0)	accept	accept
(0,0)	(a,0)	null or reject	accept
(a,0)	(a,0)	accept	accept
(x,0)	(a,0)	null or reject	accept
(x,y)	(a,0)	null or reject	accept
(a,y)	(a,0)	null or reject	accept
(0,y)	(a,0)	null or reject	null or reject
DBCS			
(0,0)	(0,0)	accept	accept
(a,0)	(0,0)	accept	accept
(x,0)	(0,0)	accept	accept
(a,b)	(0,0)	accept	accept
(x,y)	(0,0)	accept	accept
(a,y)	(0,0)	accept	accept
(x,b)	(0,0)	accept	accept
(0,b)	(0,0)	accept	accept
(0,y)	(0,0)	accept	accept

PC Version of IXF File Format

Table 16 (Page 2 of 2). Summary of Import Utility Code Page Semantics (Existing Table). This table assumes there is no conversion table between a and x.

CODE PAGE ATTRIBUTES OF PC/IXF DATA TYPE	CODE PAGE ATTRIBUTES OF TARGET DATABASE COLUMN	RESULTS OF IMPORT	
		Without FORCEIN	With FORCEIN
(0,0)	(a,b)	null or reject	accept
(a,0)	(a,b)	accept	accept
(x,0)	(a,b)	null or reject	accept
(a,b)	(a,b)	accept	accept
(x,y)	(a,b)	null or reject	accept
(a,y)	(a,b)	null or reject	accept
(x,b)	(a,b)	null or reject	accept
(0,b)	(a,b)	null or reject	null or reject
(0,y)	(a,b)	null or reject	null or reject
(0,0)	(-,b)	null or reject	accept
(a,0)	(-,b)	null or reject	null or reject
(x,0)	(-,b)	null or reject	null or reject
(a,b)	(-,b)	null or reject	null or reject
(x,y)	(-,b)	null or reject	null or reject
(a,y)	(-,b)	null or reject	null or reject
(x,b)	(-,b)	null or reject	null or reject
(0,b)	(-,b)	accept	accept
(0,y)	(-,b)	null or reject	accept

Notes:

1. See the notes for Table 15 on page 465.
2. The null or reject result is a reflection of the rules for invalid or incompatible data types (see "General Rules Governing PC/IXF File Import into Databases" on page 459).

FORCEIN Data Type Semantics

The FORCEIN option permits import of certain PC/IXF columns into target database columns of unequal and otherwise incompatible data types. The following data type semantics apply when using the FORCEIN option in either an SBCS or a DBCS environment (except where noted):

- In SBCS environments, the FORCEIN option permits import of:
 - A PC/IXF BIT data type (IXFCSBCP = 0 = IXFCDBCP for a PC/IXF character column) into a database character column (nonzero SBCS CPGID, and DBCS CPGID = 0); existing tables only

PC Version of IXF File Format

- A PC/IXF MIXED data type (nonzero IXFCSBCP and IXFDCBCP) into a database character column; both new and existing tables
- A PC/IXF GRAPHIC data type into a database FOR BIT DATA column (SBCS CPGID = 0 = DBCS CPGID); new tables only (this is always permitted for existing tables).
- The FORCEIN option does not extend the scope of valid PC/IXF data types. PC/IXF columns with data types not defined as valid in “PC/IXF Data Types” on page 449 are invalid for import with or without the FORCEIN option.
- In DBCS environments, the FORCEIN option permits import of:
 - A PC/IXF BIT data type into a database character column
 - A PC/IXF BIT data type into a database graphic column; however, if the PC/IXF BIT column is of fixed length, that length must be even. A fixed length PC/IXF BIT column of odd length is not compatible with a database graphic column. A varying-length PC/IXF BIT column *is* compatible whether its length is odd or even, although an odd-length value from a varying-length column is an invalid value for import into a database graphic column
 - A PC/IXF MIXED data type into a database character column.

Table 17 summarizes PC/IXF file import into new or existing database tables with the FORCEIN option.

Table 17 (Page 1 of 2). Summary of PC/IXF File Import with FORCEIN Option												
PC/IXF COLUMN DATA TYPE	DATABASE COLUMN DATA TYPE											
	NUMERIC					CHARACTER			GRAPH	DATETIME		
	SMALL INT	INT	BIGINT	DEC	FLT	(0,0)	(SBCS, 0) ^e	(SBCS, DBCS) ^b	b	DATE	TIME	TIME STAMP
Numeric												
-SMALLINT	N											
	E	E	E	E ^a	E							
-INTEGER		N										
	E ^a	E	E	E ^a	E							
-BIGINT			N									
	E ^a	E ^a	E	E ^a	E							
-DECIMAL				N								
	E ^a	E ^a	E ^a	E ^a	E							
-FLOAT					N							
	E ^a	E ^a	E ^a	E ^a	E							
Character												
-(0,0)						N						
						E	E w/F	E w/F	E w/F	E ^c	E ^c	E ^c
-(SBCS,0)							N	N				
						E	E	E		E ^c	E ^c	E ^c

PC Version of IXF File Format

Table 17 (Page 2 of 2). Summary of PC/IXF File Import with FORCEIN Option

PC/IXF COLUMN DATA TYPE	DATABASE COLUMN DATA TYPE											
	NUMERIC					CHARACTER			GRAPH	DATETIME		
	SMALL INT	INT	BIGINT	DEC	FLT	(0,0)	(SBCS, 0) ^e	(SBCS, DBCS) ^b	b	DATE	TIME	TIME STAMP
-(SBCS, DBCS)							N w/F ^d	N		E ^c	E ^c	E ^c
						E	E w/F	E				
Graphic												
						N w/F ^d			N			
						E			E			
Datetime												
-DATE										N		
										E		
-TIME											N	
											E	
-TIME STAMP												N
												E
<p>Note: If a PC/IXF column can be imported into a database column only with the FORCEIN option, the string 'w/F' is displayed together with an 'N' or an 'E'. An 'N' indicates that the utility is creating a new database table; an 'E' indicates that the utility is importing data to an existing database table. The FORCEIN option affects compatibility of character and graphic data types only.</p> <p>^a Individual values are rejected if they are out of range for the target numeric data type.</p> <p>^b Data type is available only in DBCS environments.</p> <p>^c Individual values are rejected if they are not valid date or time values.</p> <p>^d Applies only if the source PC/IXF data type is not supported by the target database.</p> <p>^e Data type is not available in DBCS environments.</p>												

Differences between Version 1 PC/IXF and Version 0 System/370 IXF

The following describes differences between Version 1 PC/IXF, used by the database manager, and Version 0 System/370 IXF, used by several host database products:

- PC/IXF files are ASCII, rather than EBCDIC oriented. PC/IXF files have significantly expanded code page identification, including new code page identifiers in the H record, and the use of actual code page values in the column descriptor records. There is also a mechanism for marking columns of character data as FOR BIT DATA. FOR BIT DATA columns are of special significance, because transforms which convert a PC/IXF file format to or from any other IXF or database file format cannot perform any code page translation on the values contained in FOR BIT DATA columns.
- Only the machine data form is permitted; that is, the IXFTFORM field must always contain the value M. Furthermore, the machine data must be in PC forms; that is,

Non-delimited ASCII (ASC) File Format

the IXFTMFRM field must contain the value PC. This means that integers, floating point numbers, and decimal numbers in data portions of PC/IXF data records must be in PC forms.

- Application (A) records are permitted anywhere after the H record in a PC/IXF file. They are not counted when the value of the IXFHHCNT field is computed.
- Every PC/IXF record begins with a record length indicator. This is a 6-byte character representation of an integer value containing the length, in bytes, of the PC/IXF record not including the record length indicator itself; that is, the total record length minus 6 bytes. The purpose of the record length field is to enable PC programs to identify record boundaries.
- To facilitate the compact storage of variable-length data, and to avoid complex processing when a field is split into multiple records, PC/IXF does not support Version 0 IXF X records, but does support D record identifiers. Whenever a variable-length field or a nullable field is the last field in a data D record, it is not necessary to write the entire maximum length of the field to the PC/IXF file.

Non-delimited ASCII (ASC) File Format

A Non-delimited ASCII (ASC) file is a sequential ASCII file with row delimiters. It can be used for data exchange with any ASCII product that has a columnar format for data, including word processors. Each ASC file is a stream of ASCII characters consisting of data values ordered by row and column. Rows in the data stream are separated by row delimiters. Each column within a row is defined by a beginning-ending location pair (specified by IMPORT parameters). Each pair represents locations within a row specified as byte positions. The first position within a row is byte position 1. The first element of each location pair is the byte on which the column begins, and the second element of each location pair is the byte on which the column ends. The columns may overlap. Every row in an ASC file has the same column definition.

An ASC file is defined by:

```
ASC file ::= Row 1 data || Row delimiter ||  
           Row 2 data || Row delimiter ||  
           .  
           .  
           .  
           Row n data  
  
Row i data ::= ASCII characters || Row delimiter  
  
Row Delimiter ::= ASCII line feed sequencea
```

^a The record delimiter is assumed to be a new line character, ASCII x0A. Data generated on OS/2 or the Windows operating system can use the carriage return/line feed 2-byte standard of 0x0D0A. Data in EBCDIC code pages should use the EBCDIC LF character (0x25) as the record delimiter (EBCDIC data can be

Non-delimited ASCII (ASC) File Format

loaded using the CODEPAGE option on the LOAD command). The record delimiter is never interpreted to be part of a field of data.

Sample ASC File

Following is an example of an ASC file. Each line ends with a line feed sequence (on OS/2 or the Windows operating system, each line ends with a carriage return/line feed sequence).

```
Smith, Bob      4973      15.46
Jones, Suzanne 12345      16.34
Williams, Sam  452123    193.78
```

Notes:

1. ASC files are assumed not to contain column names.
2. Character strings are *not* enclosed by delimiters. The data type of a column in the ASC file is determined by the data type of the target column in the database table.
3. A null is imported into a nullable database column if:
 - A field of blanks is targeted for a numeric, DATE, TIME, or TIMESTAMP database column
 - A location with no beginning and ending location pairs is specified
 - A location pair with beginning and ending locations equal to zero is specified
 - A row of data is too short to contain a valid value for the target column.
4. If the target column is not nullable, an attempt to import a field of blanks into a numeric, DATE, TIME, or TIMESTAMP column causes the row to be rejected.
5. If the input data is not compatible with the target column, and that column is nullable, a null is imported or the row is rejected, depending on where the error is detected. If the column is not nullable, the row is rejected. Messages are written to the message file, specifying incompatibilities that are found.

ASC Data Type Descriptions

Table 18 (Page 1 of 4). Acceptable Data Type Forms for the ASC File Format

Data Type	Form Acceptable to the Import Utility
BIGINT	<p>A constant in any numeric type (SMALLINT, INTEGER, BIGINT, DECIMAL, or FLOAT) is accepted. Individual values are rejected if they are not in the range -9 223 372 036 854 775 808 to 9 223 372 036 854 775 807. Decimal numbers are truncated to integer values. A comma, period, or colon is considered to be a decimal point. Thousands separators are not allowed.</p> <p>The beginning and ending locations should specify a field whose width does not exceed 50 bytes. Integers, decimal numbers, and the mantissas of floating point numbers can have no more than 31 digits. Exponents of floating point numbers can have no more than 3 digits.</p>

Non-delimited ASCII (ASC) File Format

<i>Table 18 (Page 2 of 4). Acceptable Data Type Forms for the ASC File Format</i>	
Data Type	Form Acceptable to the Import Utility
BLOB/CLOB	A string of characters. The character string is truncated on the right, if necessary, to match the maximum length of the target column. If the ASC <i>truncate blanks</i> option is in effect, trailing blanks are stripped from the original or the truncated string.
BLOB_FILE, CLOB_FILE, DBCLOB_FILE (DBCS only)	A delimited or non-delimited name of the file that holds the data.
CHAR	A string of characters. The character string is truncated or padded with spaces on the right, if necessary, to match the width of the target column.
DATE	A character string representing a date value in a format consistent with the country code of the target database. The beginning and ending locations should specify a field width that is within the range for the external representation of a date.
DBCLOB (DBCS only)	A string of an even number of bytes. A string of an odd number of bytes is invalid and is not accepted. A valid string is truncated on the right, if necessary, to match the maximum length of the target column.
DECIMAL	A constant in any numeric type (SMALLINT, INTEGER, BIGINT, DECIMAL, or FLOAT) is accepted. Individual values are rejected if they are not in the range of the database column into which they are being imported. If the input value has more digits after the decimal point than the scale of the database column, the excess digits are truncated. A comma, period, or colon is considered to be a decimal point. Thousands separators are not allowed. The beginning and ending locations should specify a field whose width does not exceed 50 bytes. Integers, decimal numbers, and the mantissas of floating point numbers can have no more than 31 digits. Exponents of floating point numbers can have no more than 3 digits.
FLOAT(long)	A constant in any numeric type (SMALLINT, INTEGER, BIGINT, DECIMAL, or FLOAT) is accepted. All values are valid. A comma, period, or colon is considered to be a decimal point. An uppercase or lowercase E is accepted as the beginning of the exponent of a FLOAT constant. The beginning and ending locations should specify a field whose width does not exceed 50 bytes. Integers, decimal numbers, and the mantissas of floating point numbers can have no more than 31 digits. Exponents of floating point numbers can have no more than 3 digits.
GRAPHIC (DBCS only)	A string of an even number of bytes. A string of an odd number of bytes is invalid and is not accepted. A valid string is truncated or padded with double-byte spaces (0x8140) on the right, if necessary, to match the maximum length of the target column.

Non-delimited ASCII (ASC) File Format

<i>Table 18 (Page 3 of 4). Acceptable Data Type Forms for the ASC File Format</i>	
Data Type	Form Acceptable to the Import Utility
INTEGER	<p>A constant in any numeric type (SMALLINT, INTEGER, BIGINT, DECIMAL, or FLOAT) is accepted. Individual values are rejected if they are not in the range -2 147 483 648 to 2 147 483 647. Decimal numbers are truncated to integer values. A comma, period, or colon is considered to be a decimal point. Thousands separators are not allowed.</p> <p>The beginning and ending locations should specify a field whose width does not exceed 50 bytes. Integers, decimal numbers, and the mantissas of floating point numbers can have no more than 31 digits. Exponents of floating point numbers can have no more than 3 digits.</p>
LONG VARCHAR	<p>A string of characters. The character string is truncated on the right, if necessary, to match the maximum length of the target column. If the ASC <i>truncate blanks</i> option is in effect, trailing blanks are stripped from the original or the truncated string.</p>
LONG VARGRAPHIC (DBCS only)	<p>A string of an even number of bytes. A string of an odd number of bytes is invalid and is not accepted. A valid string is truncated on the right, if necessary, to match the maximum length of the target column.</p>
SMALLINT	<p>A constant in any numeric type (SMALLINT, INTEGER, BIGINT, DECIMAL, or FLOAT) is accepted. Individual values are rejected if they are not in the range -32 768 to 32 767. Decimal numbers are truncated to integer values. A comma, period, or colon is considered to be a decimal point. Thousands separators are not allowed.</p> <p>The beginning and ending locations should specify a field whose width does not exceed 50 bytes. Integers, decimal numbers, and the mantissas of floating point numbers can have no more than 31 digits. Exponents of floating point numbers can have no more than 3 digits.</p>
TIME	<p>A character string representing a time value in a format consistent with the country code of the target database.</p> <p>The beginning and ending locations should specify a field width that is within the range for the external representation of a time.</p>
TIMESTAMP	<p>A character string representing a time stamp value acceptable for storage in a database.</p> <p>The beginning and ending locations should specify a field width that is within the range for the external representation of a time stamp.</p>
VARCHAR	<p>A string of characters. The character string is truncated on the right, if necessary, to match the maximum length of the target column. If the ASC <i>truncate blanks</i> option is in effect, trailing blanks are stripped from the original or the truncated string.</p>

Non-delimited ASCII (ASC) File Format

Table 18 (Page 4 of 4). Acceptable Data Type Forms for the ASC File Format

Data Type	Form Acceptable to the Import Utility
VARGRAPHIC (DBCS only)	A string of an even number of bytes. A string of an odd number of bytes is invalid and is not accepted. A valid string is truncated on the right, if necessary, to match the maximum length of the target column.

Non-delimited ASCII (ASC) File Format

Appendix D. How the DB2 Library Is Structured

The DB2 Universal Database library consists of SmartGuides, online help, and books. This section describes the information that is provided, and how to access it.

To access product information online, you can use the Information Center. You can view task information, DB2 books, troubleshooting information, sample programs, and DB2 information on the Web. See "Information Center" on page 486 for details.

SmartGuides

SmartGuides help you complete some administration tasks by taking you through each task one step at a time. SmartGuides are available through the Control Center. The following table lists the SmartGuides.

Note: Not all SmartGuides are available for the partitioned database environment.

SmartGuide	Helps you to...	How to Access...
<i>Add Database</i>	Catalog a database on a client workstation.	From the Client Configuration Assistant, click on Add .
<i>Create Database</i>	Create a database, and perform some basic configuration tasks.	From the Control Center, click with the right mouse button on the Databases icon and select Create->New .
<i>Performance Configuration</i>	Tune the performance of a database by updating configuration parameters to match your business requirements.	From the Control Center, click with the right mouse button on the database you want to tune and select Configure performance .
<i>Backup Database</i>	Determine, create, and schedule a backup plan.	From the Control Center, click with the right mouse button on the database you want to backup and select Backup->Database using SmartGuide .
<i>Restore Database</i>	Recover a database after a failure. It helps you understand which backup to use, and which logs to replay.	From the Control Center, click with the right mouse button on the database you want to restore and select Restore->Database using SmartGuide .
<i>Create Table</i>	Select basic data types, and create a primary key for the table.	From the Control Center, click with the right mouse button on the Tables icon and select Create->Table using SmartGuide .
<i>Create Table Space</i>	Create a new table space.	From the Control Center, click with the right mouse button on the Table spaces icon and select Create->Table space using SmartGuide .

Online Help

Online help is available with all DB2 components. The following table describes the various types of help. You can also access DB2 information through the Information Center. For information see “Information Center” on page 486.

Type of Help	Contents	How to Access...
<i>Command Help</i>	Explains the syntax of commands in the command line processor.	From the command line processor in interactive mode, enter: <i>? command</i> where <i>command</i> is a keyword or the entire command. For example, ? catalog displays help for all the CATALOG commands, while ? catalog database displays help for the CATALOG DATABASE command.
<i>Control Center Help</i>	Explains the tasks you can perform in a window or notebook. The help includes prerequisite information you need to know, and describes how to use the window or notebook controls.	From a window or notebook, click on the Help push button or press the F1 key.
<i>Message Help</i>	Describes the cause of a message, and any action you should take.	From the command line processor in interactive mode, enter: <i>? XXXnnnnn</i> where <i>XXXnnnnn</i> is a valid message identifier. For example, ? SQL30081 displays help about the SQL30081 message. To view message help one screen at a time, enter: <i>? XXXnnnnn more</i> To save message help in a file, enter: <i>? XXXnnnnn > filename.ext</i> where <i>filename.ext</i> is the file where you want to save the message help.
<i>SQL Help</i>	Explains the syntax of SQL statements.	From the command line processor in interactive mode, enter: help statement where <i>statement</i> is an SQL statement. For example, help SELECT displays help about the SELECT statement.

Type of Help	Contents	How to Access...
<i>SQLSTATE Help</i>	Explains SQL states and class codes.	From the command line processor in interactive mode, enter: <i>? sqlstate</i> or <i>? class-code</i> where <i>sqlstate</i> is a valid five-digit SQL state and the <i>class-code</i> is first two digits of the SQL state. For example, ? 08003 displays help for the 08003 SQL state, while ? 08 displays help for the 08 class code.

DB2 Books

The table in this section lists the DB2 books. They are divided into two groups:

Cross-platform books These books contain the common DB2 information for UNIX-based and Intel-based platforms.

Platform-specific books These books are for DB2 on a specific platform. For example, for DB2 on OS/2, on Windows NT, and on the UNIX-based platforms, there are separate *Quick Beginnings* books.

Most books are available in HTML and PostScript format, and in hardcopy that you can order from IBM. The exceptions are noted in the table.

If you want to read the English version of the books, they are always provided in the directory that contains the English documentation.

You can obtain DB2 books and access information in a variety of different ways:

View	See "Viewing Online Books" on page 483.
Search	See "Searching Online Books" on page 484.
Print	See "Printing the PostScript Books" on page 484.
Order	See "Ordering the Printed DB2 Books" on page 485.

Book Name	Book Description	Form Number File Name
Cross-Platform Books		
<i>Administration Getting Started</i>	Introduces basic DB2 database administration concepts and tasks, and walks you through the primary administrative tasks.	S10J-8154 db2k0x50
<i>Administration Guide</i>	Contains information required to design, implement, and maintain a database to be accessed either locally or in a client/server environment.	S10J-8157 db2d0x51
<i>API Reference</i>	Describes the DB2 application programming interfaces (APIs) and data structures you can use to manage your databases. Explains how to call APIs from your applications.	S10J-8167 db2b0x51

Book Name	Book Description	Form Number File Name
<i>CLI Guide and Reference</i>	Explains how to develop applications that access DB2 databases using the DB2 Call Level Interface, a callable SQL interface that is compatible with the Microsoft ODBC specification.	S10J-8159 db2l0x50
<i>Command Reference</i>	Explains how to use the command line processor, and describes the DB2 commands you can use to manage your database.	S10J-8166 db2n0x51
<i>DB2 Connect Enterprise Edition Quick Beginnings</i>	Provides planning, migrating, installing, configuring, and using information for DB2 Connect Enterprise Edition. Also contains installation and setup information for all supported clients.	S10J-7888 db2cyx51
<i>DB2 Connect Personal Edition Quick Beginnings</i>	Provides planning, installing, configuring, and using information for DB2 Connect Personal Edition.	S10J-8162 db2c1x51
<i>DB2 Connect User's Guide</i>	Provides concepts, programming and general using information about the DB2 Connect products.	S10J-8163 db2c0x51
<i>DB2 Connectivity Supplement</i>	Provides setup and reference information for customers who want to use DB2 for AS/400, DB2 for OS/390, DB2 for MVS, or DB2 for VM as DRDA Application Requesters with DB2 Universal Database servers, and customers who want to use DRDA Application Servers with DB2 Connect (formerly DDCS) application requesters. Note: Available in HTML and PostScript formats only.	No form number db2h1x51
<i>Embedded SQL Programming Guide</i>	Explains how to develop applications that access DB2 databases using embedded SQL, and includes discussions about programming techniques and performance considerations.	S10J-8158 db2a0x50
<i>Glossary</i>	Provides a comprehensive list of all DB2 terms and definitions. Note: Available in HTML format only.	No form number db2t0x50
<i>Installing and Configuring DB2 Clients</i>	Provides installation and setup information for all DB2 Client Application Enablers and DB2 Software Developer's Kits. Note: Available in HTML and PostScript formats only.	No form number db2iyx51
<i>Master Index</i>	Contains a cross reference to the major topics covered in the DB2 library. Note: Available in PostScript format and hardcopy only.	S10J-8170 db2w0x50
<i>Messages Reference</i>	Lists messages and codes issued by DB2, and describes the actions you should take.	S10J-8168 db2m0x51

Book Name	Book Description	Form Number File Name
<i>DB2 Replication Guide and Reference</i>	Provides planning, configuring, administering, and using information for the IBM Replication tools supplied with DB2.	S95H-0999 db2e0x52
<i>Road Map to DB2 Programming</i>	Introduces the different ways your applications can access DB2, describes key DB2 features you can use in your applications, and points to detailed sources of information for DB2 programming.	S10J-8155 db2u0x50
<i>SQL Getting Started</i>	Introduces SQL concepts, and provides examples for many constructs and tasks.	S10J-8156 db2y0x50
<i>SQL Reference</i>	Describes SQL syntax, semantics, and the rules of the language. Also includes information about release-to-release incompatibilities, product limits, and catalog views.	S10J-8165 db2s0x51
<i>System Monitor Guide and Reference</i>	Describes how to collect different kinds of information about your database and the database manager. Explains how you can use the information to understand database activity, improve performance, and determine the cause of problems.	S10J-8164 db2f0x50
<i>Troubleshooting Guide</i>	Helps you determine the source of errors, recover from problems, and use diagnostic tools in consultation with DB2 Customer Service.	S10J-8169 db2p0x50
<i>What's New</i>	Describes the new features, functions, and enhancements in DB2 Universal Database, Version 5.2, including information about Java-based tools.	S04L-6230 db2q0x51
Platform-Specific Books		
<i>Building Applications for UNIX Environments</i>	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a UNIX system.	S10J-8161 db2axx51
<i>Building Applications for Windows and OS/2 Environments</i>	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a Windows or OS/2 system.	S10J-8160 db2a1x50
<i>DB2 Personal Edition Quick Beginnings</i>	Provides planning, installing, migrating, configuring, and using information for DB2 Universal Database Personal Edition on OS/2, Windows 95, and the Windows NT operating systems.	S10J-8150 db2i1x50
<i>DB2 SDK for Macintosh Building Your Applications</i>	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a Macintosh system. Note: Available in PostScript format and hardcopy for DB2 Version 2.1.2 only.	S50H-0528 sqla7x02
<i>DB2 SDK for SCO OpenServer Building Your Applications</i>	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a SCO OpenServer system. Note: Available for DB2 Version 2.1.2 only.	S89H-3242 sqla9x02

Book Name	Book Description	Form Number File Name
<i>DB2 SDK for SINIX Building Your Applications</i>	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a SINIX system. Note: Available in PostScript format and hardcopy for DB2 Version 2.1.2 only.	S50H-0530 sqla8x00
<i>Quick Beginnings for OS/2</i>	Provides planning, installing, migrating, configuring, and using information for DB2 Universal Database on OS/2. Also contains installing and setup information for all supported clients.	S10J-8147 db2i2x50
<i>Quick Beginnings for UNIX</i>	Provides planning, installing, configuring, migrating, and using information for DB2 Universal Database on UNIX-based platforms. Also contains installing and setup information for all supported clients.	S10J-8148 db2ixx51
<i>Quick Beginnings for Windows NT</i>	Provides planning, installing, configuring, migrating, and using information for DB2 Universal Database on the Windows NT operating system. Also contains installing and setup information for all supported clients.	S10J-8149 db2i6x50
<i>DB2 Extended Enterprise Edition for UNIX Quick Beginnings</i>	Provides planning, installing, configuring, and using information for DB2 Universal Database Extended Enterprise Edition for UNIX. This book supercedes the <i>DB2 Extended Enterprise Edition Quick Beginnings for AIX</i> book, and is suitable for use with all versions of DB2 Extended Enterprise Edition that run on UNIX-based platforms.	S99H-8314 db2v3x51
<i>DB2 Extended Enterprise Edition for Windows NT Quick Beginnings</i>	Provides planning, installing, configuring, and using information for DB2 Universal Database Extended Enterprise Edition for Windows NT.	S09L-6713 db2v6x51

Notes:

1. The character in the sixth position of the file name indicates the language of a book. For example, the file name db2d0e50 indicates that the *Administration Guide* is in English. The following letters are used in the file names to indicate the language of a book:

Language	Identifier	Language	Identifier
Brazilian Portuguese	B	Japanese	J
Bulgarian	U	Korean	K
Czech	X	Norwegian	N
Danish	D	Polish	P
English	E	Russian	R
Finnish	Y	Simp. Chinese	C
French	F	Slovenia	L
German	G	Spanish	Z
Greek	A	Swedish	S
Hungarian	H	Trad. Chinese	T

Italian

I

Turkish

M

2. For late breaking information that could not be included in the DB2 books:
 - On UNIX-based platforms, see the Release.Notes file. This file is located in the DB2DIR/Readme/%L directory, where %L is the locale name and DB2DIR is:
 - /usr/lpp/db2_05_00 on AIX
 - /opt/IBMdb2/V5.0 on HP-UX, Solaris, SCO UnixWare 7, and SGI.
 - On other platforms, see the RELEASE.TXT file. This file is located in the directory where the product is installed.

Viewing Online Books

The manuals included with this product are in Hypertext Markup Language (HTML) softcopy format. Softcopy format enables you to search or browse the information, and provides hypertext links to related information. It also makes it easier to share the library across your site.

You can use any HTML Version 3.2-compliant browser to view the online books.

To view online books:

- If you are running DB2 administration tools, use the Information Center. See “Information Center” on page 486 for details.
- Use the open file function of your Web browser. The page you open contains descriptions of and links to DB2 books:
 - On UNIX-based platforms, open the following page:
`file:/INSTHOME/sql1lib/doc/%L/html/index.htm`
where %L is the locale name.
 - On other platforms, open the following page:
`sql1lib\doc\html\index.htm`

The path is located on the drive where DB2 is installed.

You can also open the page by double-clicking on the **DB2 Online Books** icon. Depending on the system you are using, the icon is in the main product folder or the Windows Start menu.

Note: The **DB2 Online Books** icon is only available if you do not install the Information Center.

Setting up a Document Server

By default the DB2 information is installed on your local system. This means that each person who needs access to the DB2 information must install the same files. To have the DB2 information stored in a single location, use the following instructions:

1. Copy all files and sub-directories from \sql1lib\doc\html on your local system to a web server. Each book has its own sub-directory containing all the necessary

HTML and GIF files that make up the book. Ensure that the directory structure remains the same.

2. Configure the web server to look for the files in the new location. For information, see *Setting up DB2 Online Documentation on a Web Server* at:

<http://www.software.ibm.com/data/pubs/papers/db2html.html>

3. If you are using the Java version of the Information Center, you can specify a base URL for all HTML files. You should use the URL for the list of books.
4. Once you are able to view the book files, you should bookmark commonly viewed topics such as:
 - List of books
 - Tables of contents of frequently used books
 - Frequently referenced articles like the *ALTER TABLE* topic
 - Search form.

For information about setting up a search, see the *What's New* book.

Searching Online Books

To search for information in the HTML books, you can do the following:

- Click on **Search the DB2 Books** at the bottom of any page in the HTML books. Use the search form to find a specific topic.
- Click on **Index** at the bottom of any page in an HTML book. Use the Index to find a specific topic in the book.
- Display the Table of Contents or Index of the HTML book, and then use the find function of the Web browser to find a specific topic in the book.
- Use the bookmark function of the Web browser to quickly return to a specific topic.
- Use the search function of the Information Center to find specific topics. See "Information Center" on page 486 for details.

Printing the PostScript Books

If you prefer to have printed copies of the manuals, you can decompress and print PostScript versions. For the file name of each book in the library, see the table in "DB2 Books" on page 479.

Note: Specify the full path name for the file you intend to print.

On OS/2 and Windows platforms:

1. Copy the compressed PostScript files to a hard drive on your system. The files have a file extension of .exe and are located in the `x:\doc\language\books\ps` directory, where `x`: is the letter representing the CD-ROM drive and *language* is the two-character country code that represents your language (for example, EN for English).
2. Decompress the file that corresponds to the book that you want. The result from this step is a printable PostScript file with a file extension of .psz.

3. Ensure that your default printer is a PostScript printer capable of printing Level 1 (or equivalent) files.
4. Enter the following command from a command line:

```
print filename.psz
```

On UNIX-based platforms:

1. Mount the CD-ROM. Refer to your *Quick Beginnings* manual for the procedures to mount the CD-ROM.
2. Change to `/cdrom/doc/%L/ps` directory on the CD-ROM, where `/cdrom` is the mount point of the CD-ROM and `%L` is the name of the desired locale. The manuals will be installed in the previously-mentioned directory with file names ending with `.ps.Z`.
3. Decompress and print the manual you require using the following command:

- For AIX:

```
zcat filename | qprt -P PSprinter_queue
```

- For HP-UX, Solaris, or SCO UnixWare 7:

```
zcat filename | lp -d PSprinter_queue
```

- For Silicon Graphics IRIX and SINIX:

```
zcat < filename | lp -d PSprinter_queue
```

where *filename* is the name of the full path name and extension of the compressed PostScript file and *PSprinter_queue* is the name of the PostScript printer queue.

For example, to print the English version of *Quick Beginnings for UNIX* on AIX, you can use the following command:

```
zcat /cdrom/doc/en/ps/db2ixe50.ps.Z | qprt -P ps1
```

Ordering the Printed DB2 Books

You can order the printed DB2 manuals either as a set, or individually. There are three sets of books available. The form number for the entire set of DB2 books is SB0F-8915-00. The form number for the set of books updated for Version 5.2 is SB0F-8921-00. The form number for the books listed under the heading "Cross-Platform Books" is SB0F-8914-00.

Note: These form numbers only apply if you are ordering books that are printed in the English language.

You can also order books individually by the form number listed in "DB2 Books" on page 479. To order printed versions, contact your IBM authorized dealer or marketing representative, or phone 1-800-879-2755 in the United States or 1-800-IBM-4YOU in Canada.

Information Center

The Information Center provides quick access to DB2 product information. You must install the DB2 administration tools to obtain the Information Center.

Depending on your system, you can access the Information Center from the:

- Main product folder
- Toolbar in the Control Center
- Windows Start menu
- Help menu of the Control Center
- **db2ic** command.

The Information Center provides the following kinds of information. Click on the appropriate tab to look at the information:

Tasks	Lists tasks you can perform using DB2.
Reference	Lists DB2 reference information, such as keywords, commands, and APIs.
Books	Lists DB2 books.
Troubleshooting	Lists categories of error messages and their recovery actions.
Sample Programs	Lists sample programs that come with the DB2 Software Developer's Kit. If the Software Developer's Kit is not installed, this tab is not displayed.
Web	Lists DB2 information on the World Wide Web. To access this information, you must have a connection to the Web from your system.

When you select an item in one of the lists, the Information Center launches a viewer to display the information. The viewer might be the system help viewer, an editor, or a Web browser, depending on the kind of information you select.

The Information Center provides some search capabilities so you can look for specific topics, and filter capabilities to limit the scope of your searches.

For a full text search, follow the *Search DB2 Books* link in each HTML file, or use the search feature of the help viewer.

The HTML search server is usually started automatically. If a search in the HTML information does not work, you may have to start the search server via its icon on the Windows or OS/2 desktop.

Refer to the release notes if you experience any other problems when searching the HTML information.

Appendix E. Notices

Any reference to an IBM licensed program in this publication is not intended to state or imply that only IBM's licensed program may be used. Any functionally equivalent product, program or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the

IBM Director of Licensing,
IBM Corporation,
500 Columbus Avenue,
Thornwood, NY, 10594
USA.

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 which has been exchanged, should contact:

IBM Canada Limited
Department 071
1150 Eglinton Ave. East
North York, Ontario
M3C 1H7
CANADA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

This publication 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.

Trademarks

The following terms are trademarks or registered trademarks of the IBM Corporation in the United States and/or other countries:

ACF/VTAM	MVS/ESA
ADSTAR	MVS/XA
AISPO	NetView
AIX	OS/400
AIXwindows	OS/390
AnyNet	OS/2
APPN	PowerPC
AS/400	QMF
CICS	RACF
C Set++	RISC System/6000
C/370	SAA
DATABASE 2	SP
DatagLANce	SQL/DS
DataHub	SQL/400
DataJoiner	S/370
DataPropagator	System/370
DataRefresher	System/390
DB2	SystemView
Distributed Relational Database Architecture	VisualAge
DRDA	VM/ESA
Extended Services	VSE/ESA
FFST	VTAM
First Failure Support Technology	WIN-OS/2
IBM	
IMS	
Lan Distance	

Trademarks of Other Companies

The following terms are trademarks or registered trademarks of the companies listed:

C-bus is a trademark of Corollary, Inc.

HP-UX is a trademark of Hewlett-Packard.

Java, HotJava, Solaris, Solstice, and Sun are trademarks of Sun Microsystems, Inc.

Microsoft, Windows, Windows NT, Visual Basic, and the Windows logo are trademarks or registered trademarks of Microsoft Corporation.

PC Direct is a trademark of Ziff Communications Company and is used by IBM Corporation under license.

SCO is a trademark of The Santa Cruz Operation.

SINIX is a trademark of Siemens Nixdorf.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Other company, product, or service names, which may be denoted by a double asterisk (**), may be trademarks or service marks of others.

Index

Special Characters

!, shell command 69
\, line continuation character 78

A

abnormal termination
 restart 360
access path
 creating new 380
 optimizing 378
action
 precompile/bind option 101, 309
ACTIVATE DATABASE 87
ADD NODE 89
admin configuration
 file 172
 network parameter values 406
 resetting to default 352
 sample 172
adsm_mgmtclass
 database configuration parameter 181
adsm_nodename
 database configuration parameter 181
adsm_owner
 database configuration parameter 181
adsm_password
 database configuration parameter 181
agent_stack_sz
 database manager configuration parameter 190
agentpri
 database manager configuration parameter 190
alias
 naming conventions 431
anyorder 293
app_ctl_heap_sz
 database configuration parameter 181
APPC node
 uncataloging 402
applheapsz
 database configuration parameter 181
application record, PC/IXF 448
ASC data type descriptions 472
ASC file
 format 471

ASC file (*continued*)
 sample 472
ASC, as an import file type 220
aslheapsz
 database manager configuration parameter 190
ATTACH 91
Audit Facility Administrator Tool 6
audit_buf_sz
 database manager configuration parameter 190
authentication
 database manager configuration parameter 190
authentication ID
 naming conventions 431
authorities
 granting when creating a database 148
authority level
 direct, defined 175
 for creating databases, granting 148
 indirect, defined 175
 report 174
Autoloader 7
autorestart
 database configuration parameter 181
avg_appls
 database configuration parameter 182

B

backbufsz
 database manager configuration parameter 190
BACKUP DATABASE 93
backup_pending
 database configuration parameter 182
Benchmark Tool 8
binarynumerics 297
BIND 98
 to create new access path 380
Bind File Description Tool 13
bindfile
 precompile option 309
binding
 defaults 110
 errors during 147
 implicitly created schema 110, 324
blocking
 precompile/bind option 102, 310

buffpage
 database configuration parameter 182

C

case sensitivity 81
 in naming conventions 431
CATALOG APPC NODE 111
CATALOG APPCLU NODE 114
CATALOG APPN NODE 116
CATALOG DATABASE 118
CATALOG DCS DATABASE 121
CATALOG GLOBAL DATABASE 124
CATALOG IPX/SPX NODE 126
CATALOG LOCAL NODE 129
CATALOG NAMED PIPE NODE 131
CATALOG NETBIOS NODE 133
CATALOG ODBC DATA SOURCE 135
CATALOG TCP/IP NODE 136
catalogcache_sz
 database configuration parameter 182
cataloging
 database 118
 host database 121
CCSIDG
 precompile/bind option 102, 310
CCSIDM
 precompile/bind option 102, 310
CCSIDS
 precompile/bind option 102, 310
CHANGE DATABASE COMMENT 139
CHANGE ISOLATION LEVEL 141
character string delimiter 435
characters, special
 permitted in CLP commands 81
chardel 165, 230, 298
charsub
 precompile/bind option 102, 310
chnpgps_thresh
 database configuration parameter 182
CLI configuration
 sample 176
CLOSE statement
 executing through the CLP 419
cnulreqd
 precompile/bind option 103, 311
codepage 295
 database configuration parameter 182
codeset
 database configuration parameter 182
coldel 165, 230, 298
collection
 precompile/bind option 103, 311
column
 naming conventions 431
column descriptor record, PC/IXF 443
column values, invalid 460
columns, incompatible 460
command line processor
 accessing databases through 69
 accessing help 70
 batch mode 69
 command mode 69
 description 69
 interactive input mode 69
 invoking 69
 quitting 69, 331
 shell command 69
 terminating 69, 398
 using 78
command syntax
 interpreting 427
compound 228
configuration, admin
 resetting to default 352
 sample 172
configuration, CLI
 sample 176
configuration, database
 resetting to default 354
 sample 179
 updating 411
configuration, database manager
 sample 188
conn_elapse
 database manager configuration parameter 190
connect
 precompile option 311
CONNECT statement
 database connection 78
 executing through the CLP 420
consistency
 required for backup 96
continuation character, line
 in command line processor 78
Control Center 14
conventions, naming
 for aliases 431
 for authentication IDs 431
 for columns 431

- conventions, naming (*continued*)
 - for database manager objects 431
 - for databases 431
 - for tables 431
 - for views 431
 - in SNA 431
- copyprotect
 - database configuration parameter 182
- country
 - database configuration parameter 182
- cpuspeed
 - database manager configuration parameter 190
- CREATE DATABASE 143
- Create Instance 32
- Create Sample Database 48
- cursor stability (CS)
 - changing 141

D

- Data Declustering Tool 53
- data fragmentation
 - eliminating, by table reorganization 342
- data integrity
 - maintaining, with isolation levels 141
- data record, PC/IXF 446
- data skew, redistributing data in nodegroup 337
- data type descriptions
 - ASC 472
 - DEL 436
 - PC/IXF 455
- data types
 - PC/IXF 449
- database
 - cataloging 118
 - changing comments in directory 139
 - checking authorizations 174
 - connection, overview of 78
 - deleting, ensuring recovery with log files 157
 - exporting table to a file 161
 - home directory entry, definition of 242
 - implicit connection 78
 - importing file to table 219
 - indirect directory entry, definition of 242
 - information 204
 - migrating, command for 303
 - monitor, resetting 358
 - naming conventions 431
 - recovering 370
 - remote directory entry, definition of 242
- database (*continued*)
 - removing 157
 - removing entries (uncataloging) 399
 - removing host DCS entries 401
 - reorganizing 345
 - restarting 360
 - restoring (rebuilding) 362
 - roll-forward recovery of 370
 - statistics 378
- database access
 - starting database manager 78
- database backup
 - history file 325
- database configuration
 - network parameter values 412
 - resetting to default 354
 - sample 179
 - updating 411
- Database Connection Services (DCS) directory
 - uncataloging entries 401
- database directories
 - changing comments 139
 - definition of 242
 - sample content of 241
- database manager
 - accessing from command prompt 1
 - instances of 200
 - monitor switches, checking 198, 201
 - starting 390
 - statistics 203
 - stopping 395
 - system commands 1
- database manager configuration
 - file 197
 - network parameter values 413
 - sample 188
- database monitor
 - description 415
- Database Pre-migration Tool 17
- database system monitor
 - GET DATABASE MANAGER MONITOR SWITCHES 198
 - GET MONITOR SWITCHES 201
 - GET SNAPSHOT 203
 - RESET MONITOR 358
 - UPDATE MONITOR SWITCHES 415
- database_consistent
 - database configuration parameter 182
- database_level
 - database configuration parameter 182

- datesiso 165, 230, 298
- datetime
 - precompile/bind option 103, 311
- db2
 - CMD description 69
 - command syntax 70
- DB2 Administration Server 3
- DB2 Connect
 - supported connections to other systems 122
- DB2 Governor 30
- DB2 Governor Log Query 31
- DB2 Interactive CLI 18
- DB2 library
 - books 479
 - Information Center 486
 - language identifier for books 482
 - late breaking information 483
 - online help 478
 - ordering printed books 485
 - printing PostScript books 484
 - searching online books 484
 - setting up document server 483
 - SmartGuides 477
 - structure of 477
 - viewing online books 483
- DB2 Profile Registry Command 50
- DB2 SQL Explain Tool 27
- DB2 SQLJ Profile Customizer 43
- DB2 SQLJ Profile Printer 45
- DB2 Statistics Extraction Tool 38
- db2admin 3
- db2adutl 5
- db2atld 7
- db2batch 8
- db2bfd 13
- db2cc 14
- db2cidmg 16
- db2ckmig 17
- db2cli 18
- db2cmd 19
- db2drdat 20
- db2empfa 22
- db2eva 23
- db2evmon 25
- db2exfmt 26
- db2expln 27
- db2flsn 28
- db2gov 30
- db2govlg 31
- db2icrt 32
- db2idrop 33
- db2ilist 34
- db2imigr 35
- db2ipxad 36
- db2iupdt 37
- db2look 38
- db2migdr 41
- db2mscs 42
- DB2OPTIONS
 - environment variable 71
- db2profc 43
- db2profp 45
- db2rbind 47
- db2sampl 48
- db2set 50
- db2split 53
- db2sql92 54
- db2start 57, 390
- db2stop 58, 395
- db2tbst 59
- db2trc 60
- db2uidl 63
- db2untag 64
- db2upd52 66
- db2vexp 67
- dbheap
 - database configuration parameter 182
- DEACTIVATE DATABASE 149
- dec
 - precompile/bind option 104, 312
- decdel
 - precompile/bind option 104, 312
- DECLARE CURSOR statement
 - executing through the CLP 420
- decplusblank 165, 230, 298
- decpt 166, 230, 298
- default
 - admin configuration, resetting to 352
 - database configuration, resetting to 354
- defer_import 231
- deferred_prepare
 - precompile option 312
- degree
 - precompile/bind option 104, 313
- DEL data type descriptions 436
- DEL file
 - format 433
 - sample 434

- delimited ASCII (DEL) file format 433
- delimiter
 - character string 435
- DEREGISTER 151
- DESCRIBE 152
- DETACH 156
- device, tape 94
- dft_account_str
 - database manager configuration parameter 191
- dft_client_adpt
 - database manager configuration parameter 191
- dft_client_comm
 - database manager configuration parameter 191
- dft_degree
 - database configuration parameter 183
- dft_extent_sz
 - database configuration parameter 183
- dft_loadrec_ses
 - database configuration parameter 183
- dft_mon_bufpool
 - database manager configuration parameter 191
- dft_mon_lock
 - database manager configuration parameter 191
- dft_mon_sort
 - database manager configuration parameter 191
- dft_mon_stmt
 - database manager configuration parameter 191
- dft_mon_table
 - database manager configuration parameter 191
- dft_mon_uow
 - database manager configuration parameter 191
- dft_prefetch_sz
 - database configuration parameter 183
- dft_queryopt
 - database configuration parameter 183
- dftdbpath
 - database manager configuration parameter 191
- diaglevel
 - database manager configuration parameter 191
- diagpath
 - database manager configuration parameter 191
- differences between PC/IXF and System/370 IXF 470
- dir_cache
 - database manager configuration parameter 191
- dir_obj_name
 - database configuration parameter 183
 - database manager configuration parameter 191
- dir_path_name
 - database manager configuration parameter 191
- dir_type
 - database manager configuration parameter 192
- directories
 - Database Connection Services (DCS), uncataloging entries 401
 - database, changing comments 139
 - deleting entries 402
 - node, removing entries from 402
 - system database, removing 399
 - uncataloging 399
- disconnect
 - precompile option 313
- disconnecting
 - command line processor front-end and back-end processes 398
- discover
 - database manager configuration parameter 192
- discover_comm
 - database manager configuration parameter 192
- discover_db
 - database configuration parameter 183
- discover_inst
 - database manager configuration parameter 192
- dl_expint
 - database configuration parameter 183
- dl_num_backup
 - database configuration parameter 183
- dl_num_copies
 - database configuration parameter 183
- dl_time_drop
 - database configuration parameter 183
- dlchktime
 - database configuration parameter 184
- dldel 165, 228, 293
- dos_rqrioblk
 - database manager configuration parameter 192
- DRDA Trace 20
- drda_heap_sz
 - database manager configuration parameter 192
- DROP DATABASE 157
- DROP NODE VERIFY 159
- dumpfile 296
- dumping a trace to file 61
- Dynamic Visual Explain 67
- dynamicrules
 - precompile/bind option 104, 313

E

ECHO 160
Enable Multi-page File Allocation 22
environment variables
 auto-commit option (-c) 72
 DB2OPTIONS 71
 display DB2 interactive prompt option (-p) 75
 display output option (-o) 74
 display SQLCODE/SQLSTATE option (-e) 73
 log commands in history file option (-l) 74
 read from input file option (-f) 73
 save all output to file option (-z) 76
 save to report file option (-r) 75
 show SQLCA data option (-a) 72
 show warning messages option (-w) 76
 statement termination character option (-t) 76
 stop execution on command error option (-s) 75
 verbose output option (-v) 76
error messages
 database configuration file 187
 dropping remote database 157
 during backup 96
 during binding 110
 invalid checksum, database configuration file 354, 412
 invalid checksum, database manager configuration file 353, 356, 406, 414
estore_seg_sz
 database configuration parameter 184
Event Analyzer 23
Event Monitor Productivity Tool 25
example
 forcein 465
explain
 bind option 105, 314
Explain Table Format Tool 26
explsnap
 precompile/bind option 105, 314
EXPORT 161
export utility file formats 433
exporting 161
 choosing file formats for 163
 database table to a file 161
 to PC/IXF format 163

F

fastparse 293
fcm_num_anchors

fcm_num_anchors (*continued*)
 database manager configuration parameter 192
fcm_num_connect
 database manager configuration parameter 192
fcm_num_rqb
 database manager configuration parameter 192
fcm_num buffers
 database manager configuration parameter 192
FETCH statement
 executing through the CLP 420
file format
 delimited ASCII (DEL) 433
 non-delimited ASCII (ASC) 471
 PC version of IXF (PC/IXF) 438
file formats
 for exporting table to file 161
 for importing file to table 220
fileserv
 database manager configuration parameter 192
Find Log Sequence Number 28
FORCE APPLICATION 169
forcein 231, 298
 code page semantics 464
 data type semantics 468
 example 465
 general semantics 464
 option 463
 summary of PC/IXF file import with 469
funcpath
 precompile/bind option 105, 315

G

generic
 precompile/bind option 106, 315
GET ADMIN CONFIGURATION 171
GET AUTHORIZATIONS 174
GET CLI CONFIGURATION 176
GET CONNECTION STATE 178
GET DATABASE CONFIGURATION 179
GET DATABASE MANAGER CONFIGURATION 188
GET DATABASE MANAGER MONITOR SWITCHES 198
GET INSTANCE 200
Get IPX/SPX Internetwork Address 36
GET MONITOR SWITCHES 201
GET SNAPSHOT 203
 effect on UPDATE MONITOR SWITCHES 415
Get Tablespace State 59

- grant
 - bind option 106
- grant_group
 - bind option 106
- grant_user
 - bind option 106

H

- header record, PC/IXF 440
- HELP 217
- host systems
 - cataloging databases located on 121
 - connections supported by DB2 Connect 122
 - removing DCS catalog entries 401
- host variables
 - not supported in command line processor 82

I

- implicit connection
 - database access 78
- implieddecimal 229, 296
- IMPORT 219
 - of PC/IXF files, with forcein 469
- import of PC/IXF files
 - data type-specific rules 461
 - general rules 459
- import utility file formats 433
- importing
 - choosing file formats for 226
 - database access through DB2 Connect 227
 - DEL files 226
 - file to database table 219
 - PC/IXF file to table 222
 - PC/IXF, multiple-part files 227
 - WSF files 226
- incompatible columns 460
- index
 - reorganization 349, 350
 - statistics 378
- indexfreespace 294
- indexixf 231
- indexrec
 - database configuration parameter 184
 - database manager configuration parameter 193
- indexschema 231
- indexsort
 - database configuration parameter 184

- indicator
 - record length 439
- indoubt transaction field, description 257
- INITIALIZE TAPE 232
- insert
 - precompile/bind option 107, 315
- Integration Exchange Format (IXF) 438
- intra_parallel
 - database manager configuration parameter 193
- invalid PC/IXF column values 460
- invalid PC/IXF data type 454
- INVOKE STORED PROCEDURE 233
- ipx_socket
 - database manager configuration parameter 193
- IPX/SPX node
 - uncataloging 402
- isolation
 - precompile/bind option 107, 316

J

- java_heap_sz
 - database manager configuration parameter 193
- jdk11_path
 - database manager configuration parameter 193

K

- keepdari
 - database manager configuration parameter 193
- keywords
 - syntax for 427

L

- langlevel
 - precompile option 316
- level
 - precompile option 317
- line continuation character
 - in command line processor 78
- LIST ACTIVE DATABASES 235
- LIST APPLICATIONS 236
- LIST COMMAND OPTIONS 238
- LIST DATABASE DIRECTORY 240
- LIST DCS APPLICATIONS 244
- LIST DCS DIRECTORY 247
- LIST DRDA INDOUBT TRANSACTIONS 249
- LIST HISTORY 251

- LIST INDOUBT TRANSACTIONS 254
- List Instances 34
- LIST NODE DIRECTORY 258
- LIST NODEGROUPS 261
- LIST NODES 263
- LIST ODBC DATA SOURCES 264
- LIST PACKAGES/TABLES 266
- LIST TABLESPACE CONTAINERS 269
- LIST TABLESPACES 271
- LOAD 276
 - remote filenames 281
- LOAD QUERY 301
- load utility file formats 433
- lobsinfile 165, 228, 294
- Local Database Directory Migration 41
- local node
 - uncataloging 402
- locklist
 - database configuration parameter 184
- locks
 - resetting maximum to default 354
- locktimeout
 - database configuration parameter 184
- log file
 - listing during roll forward 373
- log_retain_status
 - database configuration parameter 185
- logbufsz
 - database configuration parameter 184
- logfilsiz
 - database configuration parameter 184
- loghead
 - database configuration parameter 184
- logpath
 - database configuration parameter 184
- logprimary
 - database configuration parameter 184
- logretain
 - database configuration parameter 184
- logsecond
 - database configuration parameter 185

M

- max_connretries
 - database manager configuration parameter 193
- max_coordagents
 - database manager configuration parameter 193
- max_idleagents 195

- max_rt_degree
 - database manager configuration parameter 194
- max_time_diff
 - database manager configuration parameter 194
- maxagents
 - database manager configuration parameter 194
- maxappls
 - database configuration parameter 185
- maxcagents
 - database manager configuration parameter 194
- maxdari
 - database manager configuration parameter 194
- maxfilop
 - database configuration parameter 185
- maxlocks
 - database configuration parameter 185
- maxtofilop
 - database manager configuration parameter 194
- messages
 - accessing help text 70
 - precompile/bind option 107, 318
- metacharacters 81
- MIGRATE DATABASE 303
- Migrate Instance 35
- min_priv_mem
 - database manager configuration parameter 194
- mincommit
 - database configuration parameter 185
- mon_heap_sz
 - database manager configuration parameter 194
- monitoring databases 198, 201
- multipage_alloc
 - database configuration parameter 185

N

- naming a binary file for output 61
- naming conventions
 - for aliases 431
 - for authentication IDs 431
 - for columns 431
 - for database manager objects 431
 - for databases 431
 - for tables 431
 - for views 431
 - in SNA 431
- NetBIOS node
 - uncataloging 402
- newlogpath
 - database configuration parameter 185

- nextactive
 - database configuration parameter 185
- nname
 - database manager configuration parameter 194
- no commit (NC)
 - changing 141
- no_type_id 228
- nochecklengths 231, 299
- node
 - directories, removing entries from 402
- node, SOCKS 137
- nodefaults 228
- nodetype
 - database manager configuration parameter 194
- nodoubledel 166, 230, 298
- noeofchar 229, 296
- noheader 294
- nolinemacro
 - precompile option 318
- non-delimited ASCII (ASC) file format 471
- norowwarnings 294
- NULL string
 - use in setting blanks 78
- nullindchar 229, 297
- num_estore_segs
 - database configuration parameter 185
- num_freqvalues
 - database configuration parameter 185
- num_initagents
 - database manager configuration parameter 194
- num_iocleaners
 - database configuration parameter 185
- num_ioservers
 - database configuration parameter 185
- num_poolagents
 - database manager configuration parameter 194
- num_quantiles
 - database configuration parameter 185
- numdb
 - database manager configuration parameter 195
- numsegs
 - database configuration parameter 186

O

- objectname
 - database manager configuration parameter 195
- Open DB2 Command Window 19
- OPEN statement
 - executing through the CLP 421

- optimization 342
- option
 - forcein 463
- optlevel
 - precompile option 318
- output
 - precompile option 318
- owner
 - precompile/bind option 107, 318

P

- package
 - force new access paths, after running statistics 380
 - precompile option 318
 - recreating 332
- packeddecimal 297
- pagefreespace 294
- parameters
 - syntax for 427
- password
 - changing through ATTACH 91
 - changing through CONNECT 422
- PC version of IXF (PC/IXF) file format 438
- PC/IXF
 - contrasted with System/370 IXF 470
 - data type descriptions 455
 - data types 449
 - invalid column values 460
 - invalid data type 454, 459
 - record types 439, 440
 - valid data type 454
- PC/IXF file
 - format 438
- PC/IXF file import
 - data type-specific rules 461
 - general rules 459
 - with forcein 469
- PC/IXF record type
 - application 448
 - column descriptor 443
 - data 446
 - header 440
 - table 441
- pckcachesz
 - database configuration parameter 186
- performance, improving 350
 - by reorganizing tables 343
- phantom quiesce 329

PRECOMPILE PROGRAM 305
 Prepare Unique Index Conversion to V5 Semantics 63
 priv_mem_thresh
 database manager configuration parameter 195
 privileges
 direct, defined 175
 granting when creating a database 148
 indirect, defined 175
 PRUNE HISTORY 325

Q

qualifier
 precompile/bind option 107, 319
 QUERY CLIENT 326
 query_heap_sz
 database manager configuration parameter 195
 queryopt
 precompile/bind option 107, 319
 quiesce
 phantom 329
 QUIESCE TABLESPACES FOR TABLE 328
 QUIT 331

R

read stability (RS)
 changing 141
 REBIND 332
 Rebind all Packages 47
 rec_his_retentn
 database configuration parameter 186
 reclen 229, 297
 RECONCILE 335
 record length indicator 439
 record type, PC/IXF
 application 448
 column descriptor 443
 data 446
 header 440
 table 441
 record types
 PC/IXF 439, 440
 recovering a database 362
 recovery
 with roll forward 370
 without roll forward 365, 366
 redirecting output 82
 REDISTRIBUTE NODEGROUP 337

REGISTER 340
 release
 database configuration parameter 186
 database manager configuration parameter 195
 precompile/bind option 108, 319
 Release Container Tag 64
 Remote Database Migration 16
 remote filenames
 LOAD 281
 remote server
 invoking stored procedure on 233
 Remove Instance 33
 REORGANIZE TABLE 342
 REORGCHK 345
 repeatable read (RR)
 changing 141
 RESET ADMIN CONFIGURATION 352
 RESET DATABASE CONFIGURATION 354
 RESET DATABASE MANAGER
 CONFIGURATION 356
 RESET MONITOR 358
 RESTART DATABASE 360
 restbufsz
 database manager configuration parameter 195
 RESTORE DATABASE 362
 restore_pending
 database configuration parameter 186
 restoring earlier versions of DB2 databases 362
 resync_interval
 database manager configuration parameter 195
 REWIND TAPE 369
 roll forward
 enabling, when backing up 96
 ROLLFORWARD DATABASE 370
 rollfwd_pending
 database configuration parameter 186
 route_obj_name
 database manager configuration parameter 195
 rrioblk
 database manager configuration parameter 196
 rules governing PC/IXF file import 459, 461
 RUNSTATS 378

S

sample ASC file 472
 sample DEL file 434
 schema
 created when creating a database 147

- SELECT statement
 - executing through the CLP 421
 - in EXPORT command 162
 - resolving ambiguous symbols in WHERE clause 81
- semantics
 - forcein, code page 464
 - forcein, data type 468
 - forcein, general 464
- seqdetect
 - database configuration parameter 186
- SET CLIENT 382
- SET RUNTIME DEGREE 385
- SET TABLESPACE CONTAINERS 387
- SET TAPE POSITION 389
- Set up Windows NT Failover Utility 42
- setting up document server 483
- sheapthresh
 - database manager configuration parameter 196
- SIGNALRM signal 394
 - starting the database manager 394
- SIGNALRM signal, starting the database manager 394
- SIGINT signal, starting database manager 394
- SIGINT signal, starting the database manager 394
- SOCKS node 137
- softmax
 - database configuration parameter 186
- sortheap
 - database configuration parameter 186
- spm_log_file_sz
 - database manager configuration parameter 196
- spm_log_path
 - database manager configuration parameter 196
- spm_max_resync
 - database manager configuration parameter 196
- spm_name
 - database manager configuration parameter 196
- SQL NULL value
 - command line processor representation 82
- SQL statements
 - accessing help 70
 - executing through the command line processor (CLP) 419
- SQL92 Compliant SQL Statement Processor 54
- sqlca
 - precompile option 319
- SQLDA structure
 - calling server procedures that use, restrictions on 233
- sqlerror
 - precompile/bind option 108, 319
- sqlflag
 - precompile option 320
- sqlrules
 - precompile option 320
- sqlstmtsz
 - database manager configuration parameter 196
- sqlwarn
 - precompile/bind option 108, 321
- ss_logon
 - database manager configuration parameter 196
- Start Control Center 14
- START DATABASE MANAGER 390
- Start DB2 57
- start_stop_time
 - database manager configuration parameter 196
- starting a trace 61
- stat_heap_sz
 - database configuration parameter 186
- statistics
 - database 378
 - database manager 203
 - reorganizing indexes 349
 - REORGCHK 345
- stmtheap
 - database configuration parameter 187
- STOP DATABASE MANAGER 395
- Stop DB2 58
- storage
 - physical 342
- stored procedure
 - invoking 233
- strdel
 - precompile/bind option 108, 321
- striptblanks 229, 298
- striptnulls 230, 298
- structure
 - delimited ASCII (DEL) file 433
 - non-delimited ASCII (ASC) file 471
- svcname
 - database manager configuration parameter 196
- syncpoint
 - precompile option 321
- syntax
 - for command line processor SQL statements 419
 - for host variables not supported in command line processor 82
 - precompile option 322
- syntax diagrams 427
- sysadm_group
 - database manager configuration parameter 196

sysctrl_group
 database manager configuration parameter 196
 sysmaint_group
 database manager configuration parameter 197
 system commands 1
 system database directory
 uncataloging 399
 System/370 IXF 470
 contrasted with PC/IXF 470

T

table
 exporting to a file 161
 importing file to 219
 naming conventions 431
 reorganization, determining if required 345
 statistics 378
 table record, PC/IXF 441
 table reorganization
 command for 342
 tape device 94
 target
 precompile option 322
 TCP/IP node
 uncataloging 402
 TERMINATE 398
 cautions on use of 398
 termination 398
 abnormal 360
 normal 396
 territory
 database configuration parameter 187
 text
 precompile/bind option 109, 323
 tm_database
 database manager configuration parameter 197
 totalfreespace 294
 tp_mon_name
 database manager configuration parameter 197
 tpname
 database manager configuration parameter 197
 Trace 60
 tracefile 20
 trust_allclnts
 database manager configuration parameter 197
 trust_clntauth
 database manager configuration parameter 197

U

udf_mem_sz
 database manager configuration parameter 197
 UNCATALOG DATABASE 399
 UNCATALOG DCS DATABASE 401
 UNCATALOG NODE 402
 UNCATALOG ODBC DATA SOURCE 404
 uncataloging
 database entries 399
 host DCS database entries 401
 system database directory 399
 uncommitted read (UR)
 changing 141
 UPDATE ADMIN CONFIGURATION 405
 Update Catalog to Support Version 5.2 66
 UPDATE CLI CONFIGURATION 407
 UPDATE COMMAND OPTIONS 409
 UPDATE DATABASE CONFIGURATION 411
 UPDATE DATABASE MANAGER
 CONFIGURATION 413
 Update Instances 37
 UPDATE MONITOR SWITCHES 415
 UPDATE RECOVERY HISTORY FILE 417
 usedefaults 229, 295
 user
 authorization 174
 user_exit_status
 database configuration parameter 187
 userexit
 database configuration parameter 187
 util_heap_sz
 database configuration parameter 187
 utility file formats 433

V

valid PC/IXF data type 454
 validate
 precompile/bind option 109, 323
 variables
 syntax for 427
 version
 precompile option 323
 view
 naming conventions 431

W

wchartype

 precompile option 323

WHERE clause

 resolving ambiguous symbols in SELECT
 statement 81

Work with ADSM Archived Images 5

workstation, remote

 cataloging databases 118

 removing catalog entries for databases from 399

 uncataloging from local workstation 402

Contacting IBM

This section lists ways you can get more information from IBM.

If you have a technical problem, please take the time to review and carry out the actions suggested by the *Troubleshooting Guide* before contacting DB2 Customer Support. Depending on the nature of your problem or concern, this guide will suggest information you can gather to help us to serve you better.

For information or to order any of the DB2 Universal Database products contact an IBM representative at a local branch office or contact any authorized IBM software remarketer.

Telephone

If you live in the U.S.A., call one of the following numbers:

- 1-800-237-5511 to learn about available service options.
- 1-800-IBM-CALL (1-800-426-2255) or 1-800-3IBM-OS2 (1-800-342-6672) to order products or get general information.
- 1-800-879-2755 to order publications.

For information on how to contact IBM outside of the United States, see Appendix A of the IBM Software Support Handbook. You can access this document by accessing the following page:

<http://www.ibm.com/support/>

then performing a search using the keyword "handbook."

Note that in some countries, IBM-authorized dealers should contact their dealer support structure instead of the IBM Support Center.

World Wide Web

<http://www.software.ibm.com/data/>

<http://www.software.ibm.com/data/db2/library/>

The DB2 World Wide Web pages provide current DB2 information about news, product descriptions, education schedules, and more. The DB2 Product and Service Technical Library provides access to frequently asked questions, fixes, books, and up-to-date DB2 technical information. (Note that this information may be in English only.)

Anonymous FTP Sites

<ftp.software.ibm.com>

Log on as anonymous. In the directory `/ps/products/db2`, you can find demos, fixes, information, and tools concerning DB2 and many related products.

Internet Newsgroups

`comp.databases.ibm-db2`, `bit.listserv.db2-l`

These newsgroups are available for users to discuss their experiences with DB2 products.

CompuServe

GO IBMDB2 to access the IBM DB2 Family forums

All DB2 products are supported through these forums.

To find out about the IBM Professional Certification Program for DB2 Universal Database, go to http://www.software.ibm.com/data/db2/db2tech/db2cert.html
--



Part Number: 04L9265

Printed in U.S.A.

S10J-8166-01



04L9265

