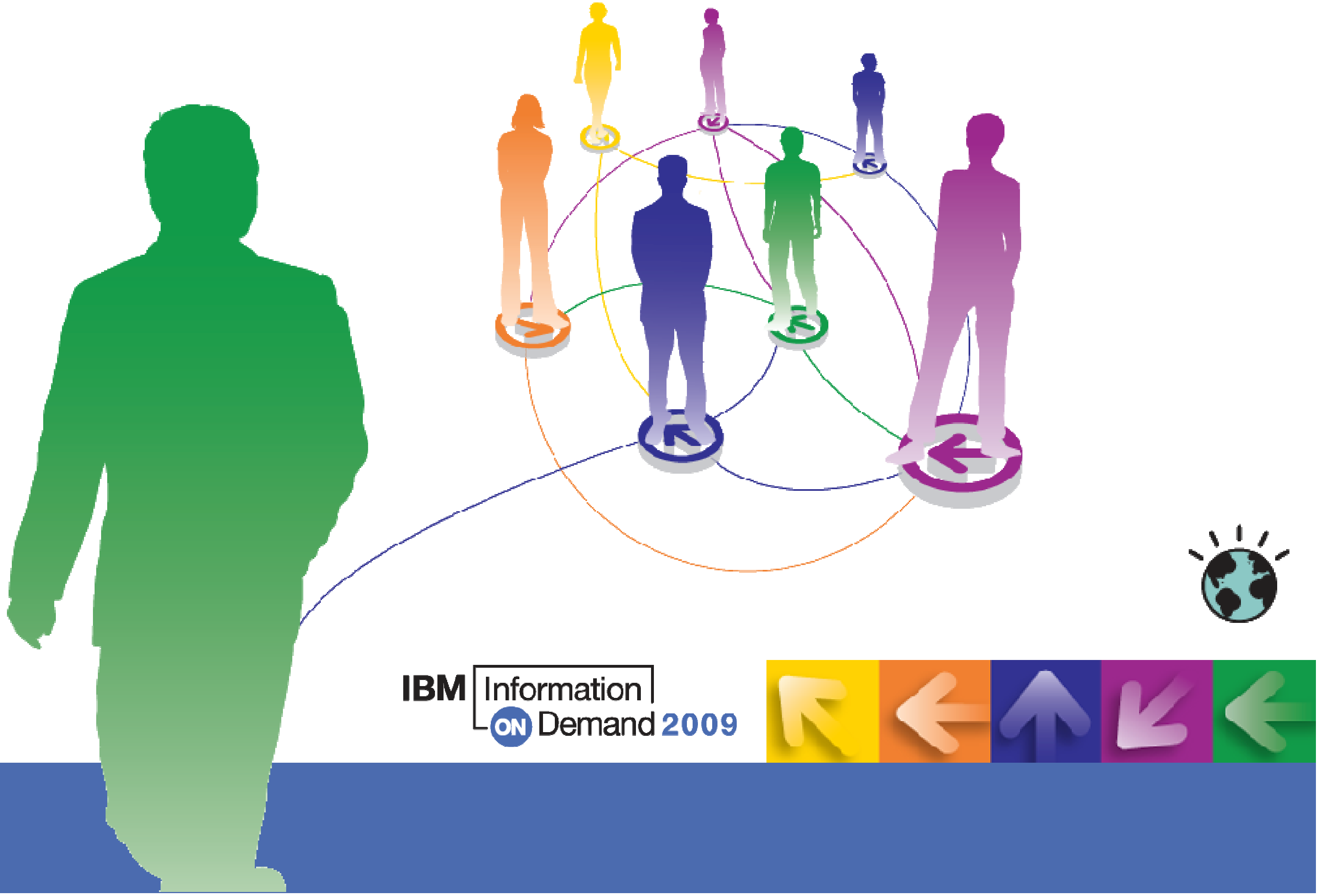
How a DB2 Performance Database can help save the day!

Mark Wilkinson, DBA, Amica Insurance (mwilkinson@amica.com)

Mary Petras, DB2 Tools Technical Support, IBM (marypetr@us.ibm.com)



IBM Information ON Demand 2009

Agenda

- General monitoring guidelines and basics
- General DB2 trace information
- Building a repository for performance data
  - Performance Database (PDB) or a Performance Warehouse (PWH)*?
  - Why? Which? How? Where? What?
- Value of a PDB and a PWH as seen from Amica Insurance
- Summary

*IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS

A Monitoring Approach: How a Monitor Can Help

- Few people have time to sit in front of a monitor and watch performance
  - Be proactive rather than reactive – use monitors to notify you when there are issues – take advantage of monitoring capabilities:
    - Capture exceptions
    - Send alerts when necessary
    - Lots of information within health graphs
- Historical Data
  - Report and analyze DB2 accounting and statistics trace data
  - Set up a PDB or a PWH to collect historical data for trending and analysis
  - Load accounting and statistics trace data into a repository

Monitoring Basics – Types of DB2 Traces

- Trace Classes
  - Accounting
  - Statistics
  - Audit
  - Monitor
  - Performance
  - Global
- Multiple trace classes per trace type
- IFCID (Instrumentation Facility Component Identifier)
  - Basic unit of reporting
  - Specific to a particular function in DB2
  - IFCID details are documented in DSNxxx.SDSNIVPD(DSNWMSGS)
- Destinations
  - SMF – Daily monitoring - default for accounting and statistics traces
  - GTF – High volume – default for performance trace
  - OPx – Buffers used by Online Monitors
  - Can be overridden by DEST parameter

Monitoring Basics – DB2 Trace Record Specifics

- Accounting and Statistics records
  - Relatively inexpensive to collect
  - Contains a wealth of information for analysis
  - Valuable information for problem determination
- Accounting Trace Records written as SMF 101
- Statistics Trace Records written as SMF 100
- Performance Trace Records written as SMF 102
  - Performance & Audit Trace Records can be expensive
  - Depends on the specific trace class and IFCID collected
- Accounting trace data is externalized when an event ends
- Statistics trace data is externalized based on user-defined time value

Monitoring Basics – DB2 Trace Records

- How to start, modify, stop traces
  - START TRACE(ABC) CLASS(1,2,3) PLAN(CUSTPLAN) DEST(SMF)
  - MODIFY TRACE(ABC) CLASS(1,2,3,7,8)
  - STOP TRACE(ABC)
- Recommended traces to start
  - Accounting Class 1, 2, 3, (7), (8), (10) (for detailed package tracing)
  - Statistics Class 1, 3, 4, 5, 6
  - DSNZPARMs: SMFACCT, SMFSTAT, STATIME, SYNCVAL
    - STATIME default is 5, set SYNCVAL to 0
    - Get statistics recorded every 5, 10, 15, etc. after the hour

Monitoring Basics – Filtering

- Filter trace data to minimize the amount of data and to reduce CPU overhead
  - Prior to DB2 9 - use PLAN, AUTHID, IFCID, or LOCATION
  - DB2 9 allows new filtering keywords with wildcarding capability that can be used in the -START TRACE for INCLUDE or EXCLUDE:
    - USERID or XUSERID: Client user ID
    - WRKSTN or XWRKSTN: Client workstation name
    - APPNAME or XAPPNAME: Client application name
    - PKGLOC or XPKGLOC: Package LOCATION name
    - PKGCOL or XPKGCOL: Package COLLECTION name
    - PKGPROG or XPKGPROG: PACKAGE name
    - CONNID or XCONNID: Connection ID
    - CORRID or XCORRID: Correlation ID
    - ROLE or XROLE: End user’s database ROLE

DB2 Trace Typical CPU Overhead

- DB2 accounting trace
  - Class 2: 1 to 10% CPU overhead (higher percentage for fetch intensive applications
  - Class 3: less than 1% CPU overhead (could be higher if latch contentions are higher)
  - Class 7 and 8 (Package level accounting): less than 5% CPU overhead
  - Class 10 (Package Detail) : higher CPU overhead and SMF volume
- DB2 monitor trace: similar to accounting
- DB2 statistics trace : negligible

DB2 Trace Records (extract from DSNWMSGS)

```
TYPE       CLASS DATA COLLECTED                                IFCIDS ACTIVATED
---------- ----- --------------------------------------------  -------------------------------
STATISTICS   1   STATISTICAL DATA                              1-2,105,106,202
             2   INSTALLATION-DEFINED STATISTICS RECORD        152
             3   DEADLOCK AND LOCK TIMEOUT INFORMATION,        172,196,250,261,262,313,258
                 CONNECT OR DISCONNECT FROM A GROUP BUFFER
                 POOL, LONG-RUNNING URS
             4   DB2 EXCEPTION CONDITIONS                      191,192,193,194,195,203,204,205,
                                                               206,207,208,209,210,235,236,
                                                               (238),267-268
             5   DATA SHARING GLOBAL  STATISTICS               230
             8   BUFFER POOL DATA SET STATISTICS               199
ACCOUNTING   1   ACCOUNTING DATA                               3,106,239
             2   IN DB2 TIME                                   232
             3   WAIT TIME FOR I/0,LOCKS, LATCHES,             6-7,8-9,32-33,44-45,(51-52,56-57)
                 DRAINS, AND CLAIMS                            117-118,127-128,170,171,174-175,
                                                               213-214,215-216,226-227,242-243
             4   INSTALLATION-DEFINED ACCOUNTING RECORD        151
             5   TIME SPENT PROCESSING IFI REQUESTS            187
             7   PACKAGE LEVEL ACCOUNTING IN-DB2 TIME          232,240
             8   PACKAGE LEVEL ACCOUNTING WAIT TIME IN DB2     6-7,8-9,32-33,44-45,(51-52,56-57)
                                                               117-118,127-128,170,171, 174-175,
                                                               213-214,215-216,226-227,241-243
AUDIT        1   AUTHORIZATION FAILURES                        140
             2   EXPLICIT GRANT AND REVOKE                     141
             3   CREATE, DROP, AND ALTER OPERATIONS AGAINST    142
                 AGAINST AUDIT TABLES
             4   FIRST CHANGE OF AUDITED OBJECT                143
             5   FIRST READ OF AUDITED OBJECT                  144
             6   SQL STATEMENT AT BIND                         145
             7   CHANGE IN AUTHORIZATION FOR AUDITED OBJECT    55,83,87,169,312
             8   UTILITY ACCESS TO ANY OBJECT                  23,24,25
             9   INSTALLATION-DEFINED AUDIT RECORD             146
```

Monitoring Basics – PDB

- Long range history (daily, weekly, monthly, ...)
  - Store and accumulate over several days, weeks, years
  - Creation and updates of tables done manually
  - Load process can be defined and scheduled using scheduling software
  - Batch reporting SAVE/CONVERT and/or FILE function provides trace data in DB2 loadable format
  - Load the data using the DB2 LOAD utility
  - DDL and DML for Performance DB tables provided in sample library
- Write your own SQL against these tables for analysis and trending
- Summarize data on weekly and monthly basis
- Data can be grouped on an interval basis

Monitoring Basics – PWH

- Long range history (daily, weekly, monthly, ...)
  - Installation option PERFORMANCEWAREHOUSE=YES
  - Automatic creation and updates of tables by PE Server
    - DDL and DML for Performance Warehouse not provided
  - Define and schedule load process from PE Client GUI
    - Batch SAVE/FILE function provides trace data in DB2 loadable format
    - DB2 load utility used to load data
- Analyze data
  - Use pre-defined ROT and SQL queries or write your own
  - Multiple reporting capabilities
  - Trend analysis available
- Supports ONLY statistics and accounting trace data

Performance Warehouse or Performance Database?

Both are a repository for performance data but there are differences:

	PDB	PWH
CREATE	MANUAL PROCESS	AUTOMATICALLY BY PE SERVER
LOAD	MANUAL PROCESS	SCHEDULED PROCESS
ACCESS	VIA SQL	VIA PE CLIENT PWH TOOL
SW REQ'S	N/A	DB2 CONNECT
TYPE OF DATA	ACCOUNTING, STATISTICS, AUDIT, SQL, etc.	ACCOUNTING & STATISTICS ONLY

Trace Output

- Gather DB2 trace information from daily SMF data
- Collect Report Data (CRD) can also help
  - Use CRD to generate appropriate DB2 trace (IFCID) data in a sequential dataset
  - Run batch reports using this dataset as input
  - Requires a PWH be installed for the DB2 subsystem in order to use CRD from PE Client for that DB2
  - If you run CRD, you can only run accounting and statistics reports from the PWH. If you collected Audit trace records, run a batch job to get the Audit report!
- A batch CRD is available – we provide a sample batch job
  - FPEZCRD located in prefix.RKO2SAMP
  - Just collects trace data – you may need to add a report step

Creating a Performance Database

- Consists of several tables logically grouped by trace types
- Before you can load these tables, the data must be made available in a non-aggregated format, compatible with the DB2 LOAD utility
- The FILE subcommand of the DB2 PE report facility generates a data set containing data in this non-aggregated format.
- The SAVE subcommand generates aggregated data and must be converted to a non-aggregated format compatible with the DB2 LOAD utility.
- The accounting and statistics Save-File utilities (see sample jobs DGOPJACO and DGOPJSCO in the prefix.SFPESAMP library) can be used for that purpose

Performance Database – Accounting Tables

- Accounting tables are related in a 1:n relationship.
- General table is the parent table - all other tables are dependent
- For each row in the General table, there may be none, one, or many rows in the other tables.
- For accounting FILE tables, the column TIMESTAMP represents the relationship and can be used for joining data.
- For accounting SAVE tables, use columns LOCAL_LOCATION to CLIENT_TRANSACTION for joining related data in different tables

Performance Database – Accounting Tables Detail

General data	One row per DB2 accounting record
Group buffer pool	One row per group buffer pool used
Package data	One row per package and DBRM executed
DDF data	One row per remote location participating in distributed activity
Buffer pool data	One row per buffer pool used
RLF data	One row per resource limit type encountered

Performance Database – Accounting Source RKO2SAMP

TYPE OF DATA	TABLE NAME	MEMBER for CREATE	DESCRIPTION	MEMBER for LOAD
General data	DB2PMFACCT_GENERAL	DGOACFGE	DGOABFGE	DGOALFGE
Group buffer pool	DB2PMFACCT_GBUFFER	DGOACFGP	DGOABFGP	DGOALFGP
Package data	DB2PMFACCT_PROGRAM	DGOACFPK	DGOABFPK	DGOALFPK
DDF data	DB2PMFACCT_DDF	DGOACFDF	DGOABFDF	DGOALFDF
Buffer pool data	DB2PMFACCT_BUFFER	DGOACFBU	DGOABFBU	DGOALFBU

Performance Database – Statistics Tables

- Statistics tables are related in a 1:n relationship.
- General table is the parent table - all other tables are dependent
- For each row in the General table, there may be none, one, or many rows in the other tables
- Use these columns for joining related data in different tables:
  - LOCAL_LOCATION
  - GROUP_NAME
  - SUBSYSTEM_ID
  - MEMBER_NAME
  - INTERVAL_TSTAMP
  - BEGIN_REC_TSTAMP

Performance Database: Statistics Tables Detail FILE Data

General data	One row for each statistics delta record, containing data from IFCID 0001 and 0002 *
Group buffer pool data	One row per group buffer pool active at the start of the corresponding delta record
DDF data	For each delta record, one row per remote location participating in distributed activity using the system-directed access method and one for all remote locations that used the application-directed access method
Buffer pool data	One row per buffer pool active at the start of the corresponding delta record
Buffer pool data set	One row for each open data set that has an I/O event rate at least one event per second during the reporting interval

* = A delta record is a set of counters describing the DB2 activity between two consecutive DB2 statistics records pairs.

Performance Database: Statistics Tables Detail SAVE Data

General data	One row for each statistics interval record, containing data from IFCID 0001 and 0002 *
Group buffer pool data	One row per group buffer pool active at the start of the corresponding interval record
DDF data	For each interval record, one row per remote location participating in distributed activity using the system-directed access method and one for all remote locations that used the application-directed access method
Buffer pool data	One row per buffer pool active at the start of the corresponding interval record
Buffer pool data set	One row for each open data set that has an I/O event rate at least one event per second during the reporting interval

* = A Statistics interval record is a set of counters describing the DB2 activity within the interval specified by the user.

Performance Database: Statistics Source in RKO2SAMP

TYPE OF DATA	TABLE NAME	MEMBER for CREATE	DESCRIPTION	MEMBER for LOAD
General data	DB2PM_STAT_GENERAL	DGOSCGEN	DGOSBGEN	DGOSLGEN
Group buffer pool data	DB2PM_STAT_GBUFFER	DGOSCGBP	DGOSBGBP	DGOSLGBP
Buffer pool data	DB2PM_STAT_BUFFER	DGOSCBUF	DGOSBBUF	DGOSLBUF
DDF data	DB2PM_STAT_DDF	DGOSCDDF	DGOSBDDF	DGOSLDDF
Buffer pool dataset	DB2PM_STAT_DATASET	DGOSCSET	DGOSBSET	DGOSLSET

Sample DB2PM Control Statements for a SAVE Data Set

```
ACCOUNTING
        REDUCE
        FROM (,00:00) TO (,23:59)
        INTERVAL (60)
        BOUNDRY (60)
  SAVE
    DATATYPE(GENERAL,DDF,PACKAGE)
        DDNAME  (ACSAVDD1)
  EXEC
```

Sample DB2PM Control Statements for a FILE Data Set

```
 ACCOUNTING
   FILE
       DATATYPE(GENERAL)
       DDNAME   (ACFILDD1)
   FILE
       DATATYPE(DDF)
       DDNAME   (ACFILDD2)
   FILE
       DATATYPE(PACKAGE)
       DDNAME   (ACFILDD3)
EXEC
```

Amica Mutual Insurance Company

Amica

AUTO HOME LIFE

✓ ***Headquarters*** One Hundred Amica Way, Lincoln, Rhode Island

✓ ***Established*** 1907—Amica is the oldest mutual insurer of automobiles in the United States.

✓ ***Company Profile*** Amica is a direct writer of personal lines insurance with Automobile, Homeowners, Personal Umbrella Liability, and Marine coverages. The company is well known in the industry for its financial strength and for providing exceptional customer service to policyholders.

✓ ***Ratings***

- A++ (Superior) from A.M. Best Company as of May 7, 2009
- Repeatedly ranked #1 for customer service as a national auto insurer by a leading consumer publication.

✓ ***Financial Strength*** Assets at year-end 2008—$3.6 billion

- Premiums earned in 2008—$1.33 billion
- Surplus at year-end 2008—$1.9 billion
- Policies in force at year-end 2008—1.2 million

Environments

✓ **Mainframes**: System z10 Business Class 2098-T04 and 2098-Q04. z/OS v1.10 operating system

✓ **DB2 z/OS** – Two production subsystems and two testing subsystems

- Production CICS – 4 TB
- Production PeopleSoft (HR, Fin) – 250 GB
- Testing CICS, Core warehouse – 2 TB
- Testing PeopleSoft – 900 GB

✓ **DB2 V9.x LUW** – IBM Content Manger, Data Warehouse – 352 GB, Billing system, etc.

✓ **Performance Database** – 16 months of Accounting and Statistics records for the two production subsystems. Located on the Testing CICS subsystem ~100 GB

Which to use –
Performance Warehouse or Performance Database

Performance Database

Pros

- ✓ Stores historical data
- ✓ Can hold reports sets for Accounting, Statistic, Audit, Locking, Record Trace, Exception and System Parameters
- ✓ You create the database objects, therefore it's customizable
- ✓ DDL is located in sample library
- ✓ You load the data. We use a mainframe scheduling product.
- ✓ Data can be aggregated or non-aggregated
- ✓ Can automate deletion of old data

Cons

- ✓ You have to write your own queries (but you can 'adapt queries' from PWH)
- ✓ You have to really understand the relationships between the tables
- ✓ You have to really understand the column definitions *
- ✓ You have to create the load jobs
- ✓ Only historical data

***Sample create table, load control and table column descriptions are located in 'The Reporting User's Guide' Chap. 19, pg 241 SC18-7979-00**

Which to use –
Performance Warehouse or Performance Database

Performance Warehouse

Pros

- ✓ Can capture current data via traces
- ✓ Can store historical data by reading SMF datasets
- ✓ Only holds report sets for Accounting and Statistics
- ✓ Can easily share queries via GUI
- ✓ A number of canned queries and rule of thumb queries provided
- ✓ Useful for debugging currently active problems (using trace & report)

Cons

- ✓ Database is automatically created by Omegamon Server, and not customizable
- ✓ Data is only loaded via Omegamon sever, can be scheduled.
- ✓ Two-step manual process to delete old data
- ✓ Separate PWH on each DB2 subsystem
- ✓ Can only select against a single schema (it's hard coded - DB2PM)

Amica's Choice:

We primarily use PDB for historical analysis and occasionally use PWH for problem analysis as it's happening.

Amica’s Performance Database - Statistics

PDB Statistics data is collected from Amica's SMF records

✓Statistics Data

- CICS and Peoplesoft production subsystems
- Collected daily
- 16-month retention period (rotating partitions)
- Summarized five minute period (zparm setting)
- Captured using the Statistics Report with the “FILE” command, then sorted to different load files and loaded directly into the Statistics tables

```
GLOBAL
   FROM (05/14/09,00:00:00.00)
  TO        (05/14/09,24:00:00.00)
STATISTICS
 FILE
   INCLUDE(SUBSYSTEMID(DSN, PSFP))
   NOEXCEPTION
   DDNAME(ACFIL01)
EXEC
```

Amica’s Performance Database - Accounting

PDB Accounting data is collected from Amica's SMF records

✓Accounting Data

- CICS production subsystem, at this time
- Collected daily, Monday through Friday, core business hours
- Summarize on the Connection Type, Correlation Name and Interval time, 15-minute intervals
- 16-month retention period (rotating partitions)
- Captured using the “REDUCE, REPORT” commands and the “SAVE” file utility
- “CONVERT” step converts the saved file (VSAM) into a sequential dataset. Then sorted to different load files and loaded directly into the Accounting tables.

```
GLOBAL
   FROM (05/14/09,08:00:00.00)          TO(05/14/09,17:15:00.00)
   INCLUDE (SUBSYSTEMID(DSN))     PRESORTED(NO)
ACCOUNTING
         REDUCE
               INTERVAL(15)  BOUNDARY(60)
         REPORT
               DDNAME(ACRPTDD) LAYOUT(SHORT)
               NOEXCEPTION          SCOPE(MEMBER)
               ORDER(CONNECT-CORRNAME-INTERVAL)
         SAVE
                DDNAME(ACSAVDD)
EXEC
```

```
//            EXEC  PGM=DGOPMICO,
//            PARM=CONVERT
```

Example 1 - ZIIP Engine Overflow

✓ **Problem** – System Administration group notice the ZIPP engine usage had jumped radically, from ~2% to 17-18% usage, overnight.

✓ **1st Analysis** – Initially we used Accounting Long reports. This proved too cumbersome. Reports had to much detail and took an hour-plus to run.

✓ **2nd Analysis** – Using QMF to query the Performance Database (PDB), we were able to quickly generate reports and graphs zeroing in on only plans that had high ZIIP engine usage. We identified multiple plans that had one module in common. The module had recently been changed.

✓ **PDB SQL**:

```
SELECT SUM(CLASS2_IIP_CPU),PLAN_NAME,DATE(INTERVAL_TIME)
FROM DB2PWH.DB2PMSACCT_GENERAL
WHERE
   SUBSYSTEM_ID ='DSN'
AND INTERVAL_TIME > '2009-05-04-09.00.00.000000'
AND INTERVAL_TIME < '2009-05-28-15.30.00.000000'
GROUP BY PLAN_NAME,DATE(INTERVAL_TIME)
ORDER BY PLAN_NAME,DATE(INTERVAL_TIME)
```

Example 1 - ZIIP Engine Overflow

- ✓ **The Explain:** Using Optimization Service Center (OSC) to explain the package SQL, we quickly identified one costly SQL statement, with a cost of 300 su. OSC showed the query was using a non-matching index scan on the second column of the index.

- ✓ **Investigation:** Using the DB2 Administration tool we were able to see that there were three indexes defined on the table, all with the same first two columns, one of which had just the first two columns. We dropped that index and created one with just the column being scanned. This dropped the cost to about eight su.

- ✓ **What Was Happening:** The query was using parallelism which caused the high number of calls and get pages

Example 1 - ZIIP Engine Overflow

✓**By the Numbers:** The execution time for the calls (according to IBM DB2 Query Monitor) averaged about .005 second; not a bad time but this module was executing about 425,000 calls per hour. After the index change it dropped to .0003 second and 76,000 calls to DB2, there by reducing the get pages from almost 50 million to 508 thousand .

Module statistics before and after (from IBM DB2 Query Monitor):

Program	Occurrences	Calls	Elapsed	Avg Elapsed	CPU	Avg CPU	GetPages
BUY1	14,814	425,704	03:23.8	0.008935	38:27.5	0.00542	49,678,590
BUY2	16,016	76,900	01:21.7	0.001062	00:21.2	0.00027	508,425

Example 2 – RID Pool Failures

✓**Adapt.** You can ‘adapt queries’ from the Performance Warehouse (PWH) and use them on your PDB. The following RID pool failure query was copied from the PWH queries and modified to evaluate our production subsystem:

```
SELECT SUM(RID_POOL_FAIL_MXLT) AS RIDPOOL_FAILURES, PLAN_NAME,
SUBSYSTEM_ID, MAINPACK
FROM DB2PWH.DB2PMSACCT_GENERAL
WHERE DATE(INTERVAL_TIME) BETWEEN '2009-05-01' AND '2009-05-31‘ AND
SUBSYSTEM_ID = 'DSN‘
GROUP BY PLAN_NAME, SUBSYSTEM_ID, MAINPACK
HAVING SUM(RID_POOL_FAIL_MXLT) > 0
ORDER BY 1 DESC
```

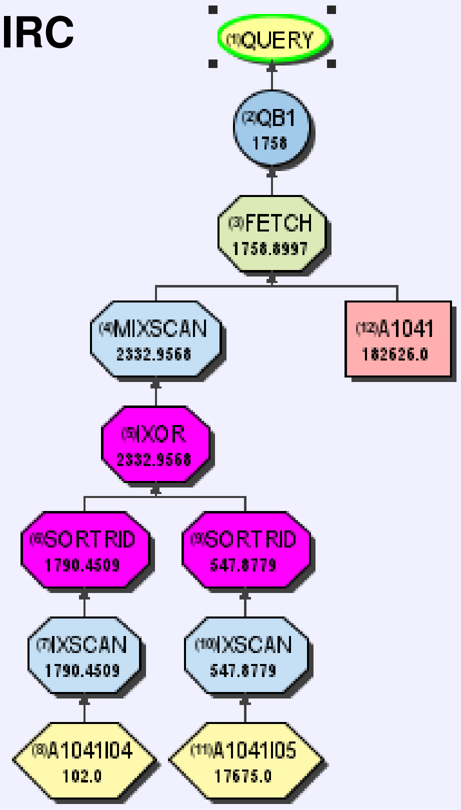
✓**Results** The number of RIDPOOL failures for the month of May is as follows:

Rid Pool Failures	Plan Name	Main Pack
3530	AR21	IRC
207	AR13	LWK
1	CT	CT

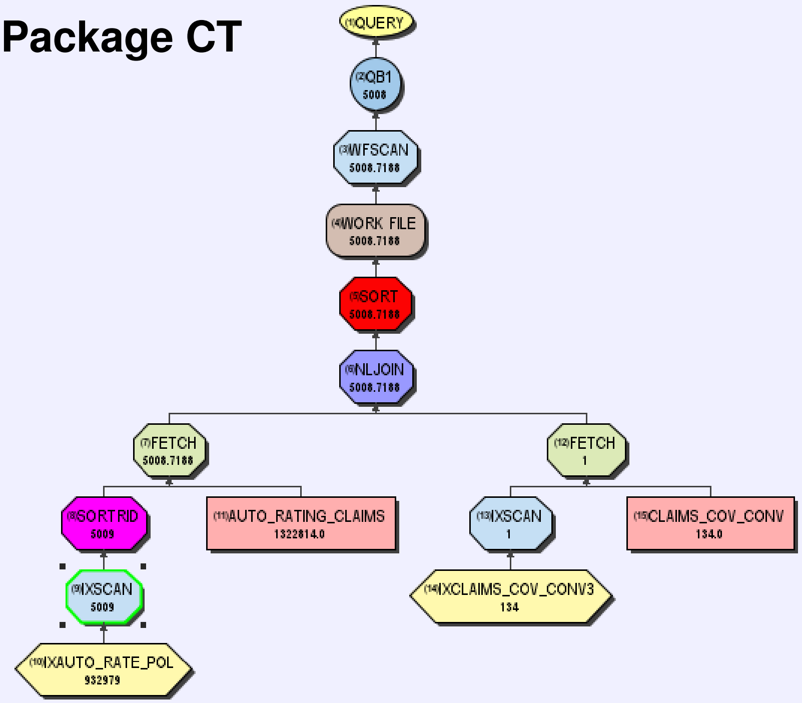
Example 2 – RID Pool Failures

✓**Analysis:** Using Optimization Service Center (OSC), we found that packages IRC and LWK were doing index OR’ing on two indexes. Package CT was sorting RIDs from an Index Scan.
We increased the size of the RIDPOOL to prevent RIDPOOL failures. The initial RIDPOOL size was approximately 27 Meg. We increase it to 54 Meg and eliminated the Rid pool failures.

Package IRC



Package CT



Example 3 – Expensive Packages

The following query shows the class 7 elapsed time by package, and the output is sorted slowest to fastest.

```
SELECT  PCK_COLLECTION_ID, PCK_ID,
            AVG(CLASS7_ELAPSED/PCK_ALLOCS_CLASS7) AS AVG_CLASS7_ELAPSED,
            SUM(PCK_ALLOCATIONS) AS NO_TIMES_PACKAGE_EXECUTED
FROM DB2PWH.DB2PMSACCT_PROGRAM
WHERE CONNECT_TYPE = 'CICS'
AND INTERVAL_TIME BETWEEN '2009-08-24-08.00.00.000000'
                                             AND '2009-08-28-16.00.00.000000'
AND PLAN_NAME IN ('AR10', 'AR24', 'AR08', 'AR13', 'AR05')
GROUP BY PCK_COLLECTION_ID,
              PCK_ID
ORDER BY 3 DESC
FETCH FIRST 20 ROWS ONLY
```

Example 3 – Expensive Packages

PCK_COLLECTION_ID	PCK_ID	AVG_CLASS 7_ELAPSED	NO_TIMES_PACKAGE_EXECUTED
BIABIAS	BX31EQU	0.69388	80312
AR13	LWK	0.50716	60
AR13	LYB	0.46990	149
AR13	LZO	0.43487	254
AR13	DWU	0.24926	19
AR13	LWR	0.23208	2
AR13	LMY	0.17670	3
AR10	IFB	0.16259	8114
AR13	RSK	0.14181	155
AR24	HIS	0.12111	1414

After evaluating the SQL in the worst packages, we discovered tablespace scans and inefficient indexes, IXORing and IXANDing with thousands of rows being merged and non-matching index scans.

Example 4 – Comparison Over Time

- ✓ The query on the next foil will compare the Class 7 elapsed times for a business week in August 2008 with a business week in August 2009.

- ✓ Percentage Change is calculated showing the difference in the elapsed time.

- ✓ This could be useful when comparing programming changes to an application or migrating to a new version of DB2.

- ✓ The query could be modified to look at different class times or to check runtimes on specific packages as changes are made.

Example 4 – Comparison Over Time

```
SELECT   PCK_COLLECTION_ID , PCK_ID
       ,  MAX(AVG_CLASS7_ELAPSED_2008) AS AVG_CLASS7_ELAPSED_2008
       ,  MAX(AVG_CLASS7_ELAPSED_2009) AS AVG_CLASS7_ELAPSED_2009
       ,  INT(MAX(AVG_CLASS7_ELAPSED_2009)/MAX(AVG_CLASS7_ELAPSED_2008) * 100) AS
PERCENT_CHANGED
FROM (SELECT  PCK_COLLECTION_ID , PCK_ID
            ,(CASE
                 WHEN   YEAR(INTERVAL_TIME) = 2009
                 THEN  AVG(CLASS7_ELAPSED/PCK_ALLOCS_CLASS7)
                 ELSE 0
              END)        AS AVG_CLASS7_ELAPSED_2009
            ,(CASE
                 WHEN   YEAR(INTERVAL_TIME) = 2008
                 THEN  AVG(CLASS7_ELAPSED/PCK_ALLOCS_CLASS7)
                 ELSE 0
              END)        AS AVG_CLASS7_ELAPSED_2008
FROM DB2PWH.DB2PMSACCT_PROGRAM
WHERE CONNECT_TYPE = 'CICS'
     AND (INTERVAL_TIME BETWEEN '2008-08-24-08.00.00.000000' AND '2008-08-28-16.00.00.000000'
     OR     INTERVAL_TIME BETWEEN '2009-08-24-08.00.00.000000' AND '2009-08-28-16.00.00.000000')
     AND PLAN_NAME IN ('AR10', 'AR24', 'AR08', 'AR13', 'AR05')
GROUP BY PCK_COLLECTION_ID,  PCK_ID        ,  YEAR(INTERVAL_TIME) ) AS TEMP
GROUP BY PCK_COLLECTION_ID,  PCK_ID
HAVING
((MAX(AVG_CLASS7_ELAPSED_2009)/MAX(AVG_CLASS7_ELAPSED_2008) * 100) > 0)  -- greater than zero shows all
packages, greater than 100 shows slower packages, between 0 and 100 shows faster packages.
AND (MAX(AVG_CLASS7_ELAPSED_2008) ) > 0  -- to eliminate divisions by zero
ORDER BY  5 DESC
FETCH FIRST 12 ROWS ONLY
```

Example 4 – Comparison Over Time

PCK COLLECTION ID	PCK_ID	AVG CLASS 7 ELAPSED 2009	AVG CLASS 7 ELAPSED 2008	PERCENT CHANGED
EDB	ERI	0.03163	0.00517	611
AR13	LWH	0.01072	0.00198	541
AR13	RLU	0.00533	0.00136	391
AR13	LMB	0.01108	0.00315	351
AR24	PYU	0.00422	0.00121	348
EDB	EHY	0.00385	0.00112	343
AR03	GTS	0.00489	0.00151	323
EDB	EIS	0.00343	0.00107	320
EDB	EHB	0.00545	0.00173	315
EDB	ERD	0.01107	0.00363	304
AR13	LMY	0.15451	0.0513	301
ACFN	BJG	0.03838	0.01304	294

Example 5 – Buffer Pool Tuning

- ✓ **Information gathering:** The two queries on the next slides were run against the PDB to gather the bufferpool hit ratios and DB2 elapsed time for our CICS production applications.

- ✓ **Frequency**: The buffer pool hit ratio from PDB was collected weekly and plotted in Microsoft Excel. We were able to gather this information and compare the bufferpool usage and DB2 elapsed times from week to week.

- ✓ **Shameless Plug for BPA:** We used the IBM Bufferpool Analysis tool to move objects to different buffer pools and we used the BPA tool to change the bufferpool sizes as well.

- ✓ **Effectiveness:** The Performance database gave us the flexibility to make changes to the buffer pools and then compare the effect of these changes from week to week.

Example 5 – Buffer Pool Tuning

```
SELECT      CASE
        WHEN CHAR(A.BP_ID) = '0' THEN 'BP0'
        WHEN CHAR(A.BP_ID) = '1' THEN 'BP1'
        WHEN CHAR(A.BP_ID) = '2' THEN 'BP2'
        WHEN CHAR(A.BP_ID) = '3' THEN 'BP3'
        WHEN CHAR(A.BP_ID) = '4' THEN 'BP4'
        WHEN CHAR(A.BP_ID) = '5' THEN 'BP5'
        WHEN CHAR(A.BP_ID) = '6' THEN 'BP6'
        WHEN CHAR(A.BP_ID) = '9' THEN 'BP9'
        WHEN CHAR(A.BP_ID) = '80' THEN 'BP32K'
        WHEN CHAR(A.BP_ID) = '81' THEN 'BP32K1'
        WHEN CHAR(A.BP_ID) = '87' THEN 'BP32K7'
        WHEN CHAR(A.BP_ID) = '121' THEN 'BP16K1'
        WHEN CHAR(A.BP_ID) = '125' THEN 'BP16K5'
    ELSE 'UNKNOWN'
END,
    A.THE_DATE,   A.SUBSYSTEM_ID, A.THE_HOUR,   AVG (CASE   WHEN A.GET_PAGE > 0
            THEN  (INT ( ( (A.GET_PAGE -  (A.SYNC_READ_IO +  A.SEQ_PREFETCH_PAGE +
                           A.LIST_PREFETCH_PAGE +   A.DYN_PREFETCH_PAGE ) ) / A.GET_PAGE) * 100) ) END)
                          AS HIT_RATIO
FROM
 (SELECT BP_ID,  DATE(BEGIN_REC_TSTAMP) AS THE_DATE,SUBSYSTEM_ID,
  HOUR (BEGIN_REC_TSTAMP) AS THE_HOUR, HPOOL_READ_SYNC, HPOOL_READ_ASYNC,
  HPOOL_WRITE_SYNC, HPOOL_WRITE_ASYNC, GET_PAGE, SYNC_READ_IO,
  SEQ_PREFETCH_PAGE, LIST_PREFETCH_PAGE, DYN_PREFETCH_PAGE
   FROM DB2PWH.DB2PM_STAT_BUFFER
   WHERE  DATE (BEGIN_REC_TSTAMP) BETWEEN '2007-01-15'  AND '2007-01-19'

GROUP BY A.BP_ID, A.THE_DATE,  A.SUBSYSTEM_ID, A.THE_HOUR
ORDER BY A.BP_ID, A.THE_DATE
```

Example 5 – Buffer Pool Tuning

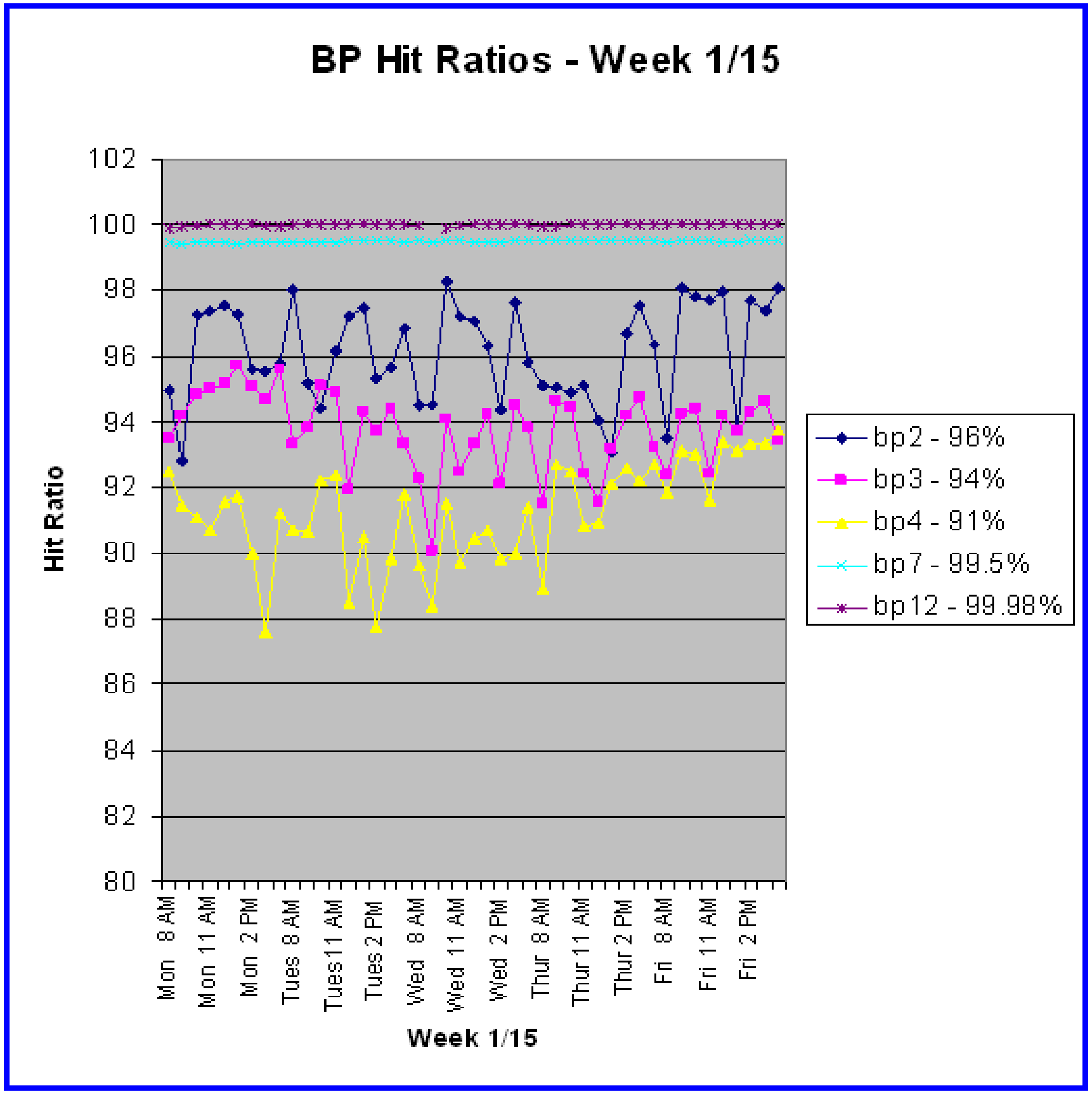
```
SELECT PLAN_NAME,
              AVG(CLASS2_ELAPSED)  AS CLASS2_ELAPSED
FROM   DB2PWH.DB2PMSACCT_GENERAL
WHERE INTERVAL_TIME BETWEEN
              '2007-01-15-08.00.00.000000' AND '2007-01-19-16.15.00.000000'
        AND PLAN_NAME IN ('AR08', 'AR10', 'AR13', 'AR24', 'LYBIAS00')
GROUP BY PLAN_NAME
ORDER BY PLAN_NAME
```

Example 5 – Buffer Pool Tuning

Week of 01/15/2009

Bufferpool Size		Avg. Hit Ratios		Plan	Tot Class 2 Elap		Avg. prior to BPA Changes
BP2	26,500	96%		AR08	0.078908	Mon.	0.030169
BP3	108,180	94%			0.027469	Wed.	
BP4	83,000	91%					
BP7	22,030	99.50%		AR10	0.017257	Mon.	0.019336
BP12	9,000	99.98%			0.017661	Wed.	
				AR13	0.024965	Mon.	0.028605
					0.027186	Wed.	
				AR24	0.044425	Mon.	0.047128
					0.038953	Wed.	
				LYBIAS00	0.038621	Mon.	0.042792
					0.030108	Wed.	

Example 5 – Buffer Pool Tuning



Summary

- A PDB gives you the opportunity to know all there is to know about your DB2 subsystems
- Often too much data to store
- Consider reducing the data on an interval basis as a single record
- Refer to these tables regularly looking for areas of improvement
- Also useful when your system is acting strange and you need to determine why
- Load the tables daily with a summarization of the activity from the previous day to reduce the amount of data kept
- Keep the data for several months or years for historical analysis
- Use the provided examples to help you solve potential performance problems

Resources and References

- PDB and PWH information
  - Chapter 19, "The Performance Database and the Performance Warehouse“, DB2 Performance Monitor for z/OS Reporting User’s Guide, SC18-7979-04
  - Redbook - IBM DB2 Performance Expert for z/OS Version 2, SG24-6867 – Chapter 5 on PDB and PWH

- General reporting information including details about the FILE and SAVE subcommands
  - IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS, IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS, Report Reference, SC19-2504

Disclaimer

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM’S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS AND/OR SOFTWARE.

IBM, the IBM logo, ibm.com, DB2, and Tivoli Omegamon Performance Expert for DB2 on z/OS are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml