IBM Software Group

The Dynamics Of Monitoring Dynamic SQL With OMEGAMON XE For DB2 PM/PE

Ed Woods

IBM Corporation

Tivoli software

e business on demand.

Agenda

- A review of dynamic SQL concepts
- Understanding the cost of dynamic SQL
- Performance and availability information available
- OMEGAMON facilities and capabilities
- Using OMEGAMON to gather and analyze Dynamic SQL performance

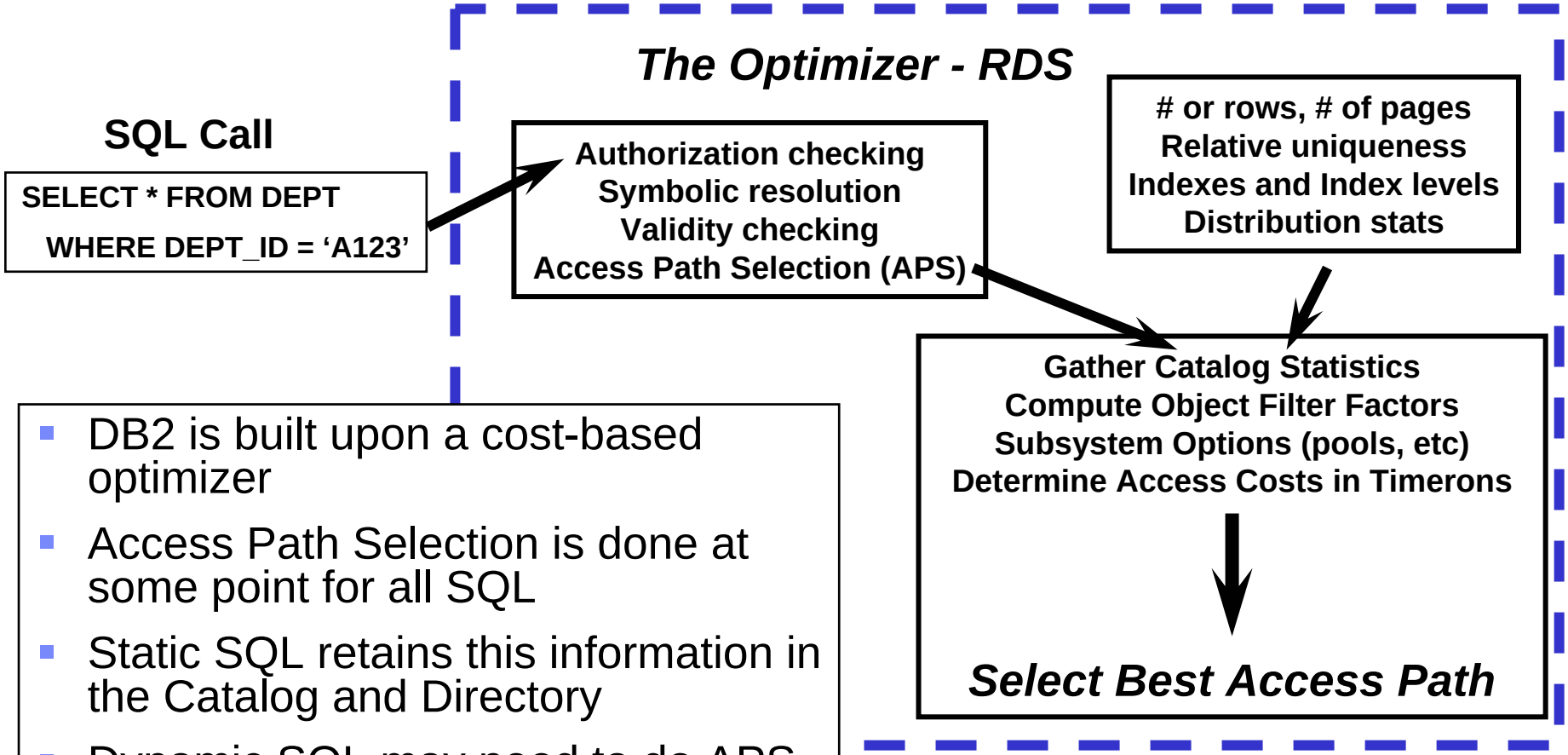
What Is Dynamic SQL?

- From an application perspective
  - Static SQL
    - SQL code hard coded into the application
  - Dynamic SQL
    - SQL text is provided by the user or generated by the application at execution time
- From a DB2 subsystem perspective
  - Static SQL
    - SQL that has been pre-compiled into a package or plan
    - May be executed directly by DB2 without additional preparation
  - Dynamic SQL
    - SQL that has not been bound before execution
    - SQL that must be prepared for execution at run time

Why Dynamic SQL?

- Application design and coding requirements
  - Application flexibility
  - Complex application requirements with multiple permutations and SQL options
- Application infrastructure requirements and considerations
  - Many application paradigms may favor dynamic SQL over static SQL
    - Example - WebSphere coding techniques such as JDBC make use of dynamic SQL
  - Some applications may require dynamic SQL

Understanding The Costs Of DB2 Access Path Selection (APS)



- DB2 is built upon a cost-based optimizer
- Access Path Selection is done at some point for all SQL
- Static SQL retains this information in the Catalog and Directory
- Dynamic SQL may need to do APS again (and again and again…….)

What Are The Costs Of Dynamic SQL?

- CPU costs
  - Cost of performing APS on an ongoing basis
  - Cost of maintaining and searching SQL cache if enabled
- Subsystem and I/O costs
  - I/O overhead on the DB2 Catalog for information needed for APS
- Memory costs
  - Memory required to cache dynamic SQL information and lessen the impact of dynamic SQL
- Analysis costs
  - Dynamic SQL will typically require somewhat different analysis and management than static SQL

Dynamic SQL Statement Caching Options

- No caching
  - Each time a statement is executed it may need to be prepared (depending upon application logic)
- Local dynamic SQL cache only
  - Local statement cache is allocated in DBM1 for each thread
  - Bind option KEEPDYNAMIC(YES)
  - Statement information is kept across commits
- Global dynamic SQL cache only
  - Maintains skeleton copies of prepared SQL to be copied to thread user copies
  - In DB2 V8 cache is allocated from storage above the 2 GB bar
  - Activate with CACHEDYN=YES option in zparms
- Full caching
  - Combination of KEEPDYNAMIC(YES) and CACHEDYN=YES

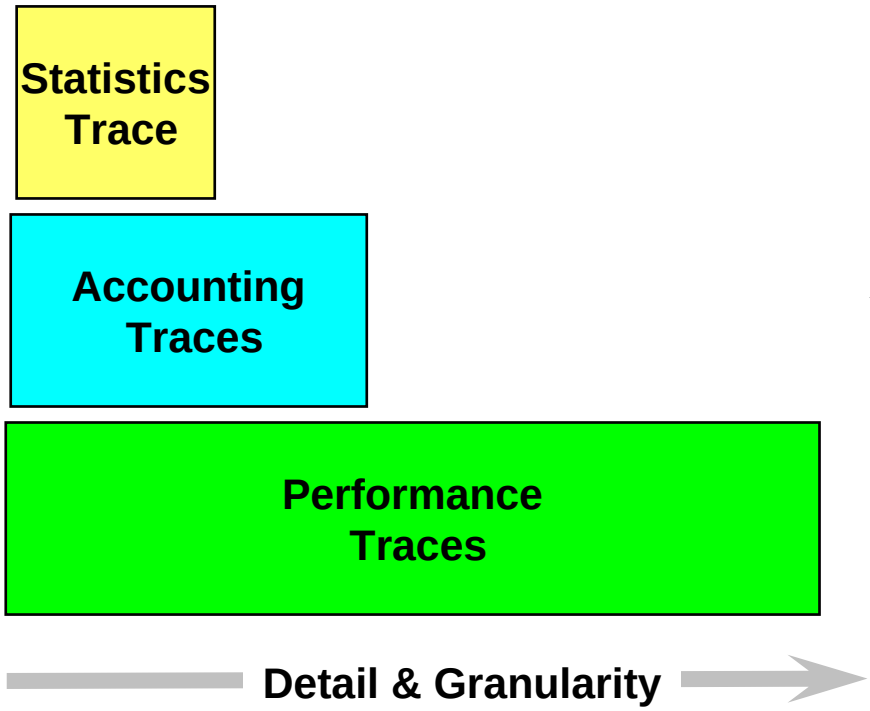
Types Of Prepares

- Full Prepare
  - Skeleton copy of the SQL is not in the cache or the cache is not active
  - Caused by a PREPARE or EXECUTE IMMEDIATE statement
- Short Prepare
  - A skeleton copy of the prepared SQL statement is copied to local storage
- Avoided Prepare
  - Prepare avoided by using full caching
  - Prepared statement information is still in thread's local storage
- Implicit Prepare
  - Due to limits such as MAXKEEPD a Prepare cannot be avoided
  - DB2 will issue the Prepare on behalf of the application

Dynamic SQL Poses Performance Analysis Challenges

- The *DYNAMIC* aspect of dynamic SQL poses additional performance analysis considerations
  - Dynamic SQL may be more of a moving target
  - May be more of a challenge to isolate and tune problem SQL
    - Potentially more permutations and combinations of SQL
- Tracing and analysis strategy for dynamic SQL based applications may be different than static SQL based applications
  - Identification and isolation of problem SQL is still key to problem analysis and tuning
  - SQL level detail may be important for a monitoring and tuning strategy
  - SQL level tracing poses challenges
    - Cost of running performance traces
    - Quantity of data gathered
    - Retention and analysis of data

Gather Information At Multiple Levels Of Detail



Subsystem level
#calls and type by time interval

Application level
#calls and type by application

Detail event level
SQL call level detail

- An effective trace gathering, retention, and analysis strategy is important
- Traces have costs, so used the appropriate tool in the right manner
- Different traces have different levels of granularity

Dynamic SQL OMEGAMON Performance Information Collection And Retention Strategies

- Statistics Traces
  - Low overhead – low volume
  - Run on an ongoing basis
  - Retention – Real time, Near Term History, SMF, snapshot
- Accounting Traces
  - High volume of data
  - Still relatively low overhead
  - Important Real time, Near Term History, SMF, snapshot
- Performance Traces
  - Low to relatively high overhead
  - Potentially high data volumes
    - Run for an interval of time
  - Retention – SMF not an option – Use the application trace facility (ATF)

OMEGAMON XE For DB2 PM/PE V4.1

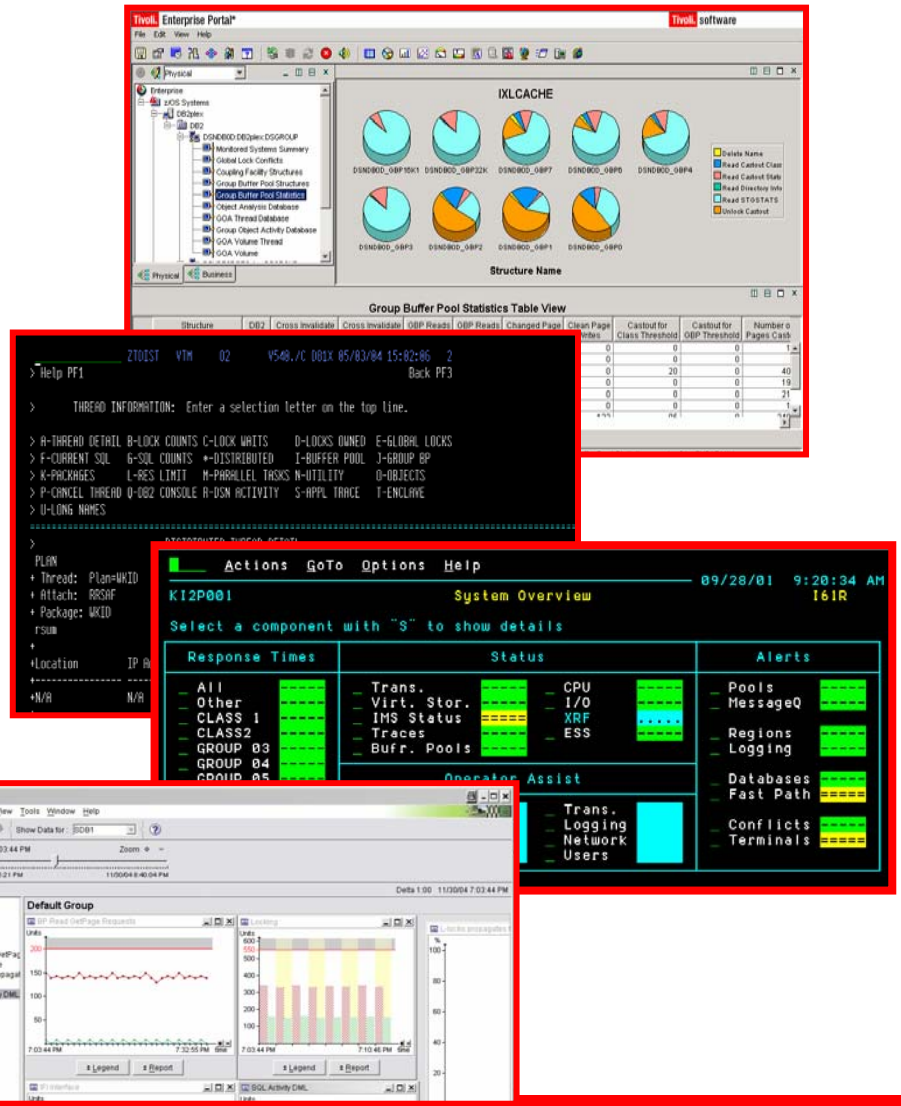
Major Features & Components

- Real Time Thread Analysis
  - ✓Thread performance
  - ✓Thread Detail
  - ✓Triggers, Procedures, & UDFs
- Real Time – DB2 subsystem
  - ✓Virtual & EDM Pool analysis
    - ✓Pool performance & snapshot analysis
  - ✓Locking & Logging Analysis
  - ✓Storage Analysis
- Application Trace Facility
  - ✓Detailed performance tracing
- Choice Of Interfaces
  - ✓(TEP, PE GUI, 3270)
- Buffer Pool Analysis (PE only)
- DB2 Connect Monitoring
- Object Analysis
  - ✓I/O & getpage analysis
  - ✓Correlate by object & App
- Locking & Lock Conflicts
- Near-Term Historical
  - ✓Near-term history online
- Historical Analysis
  - ✓Batch reporting
  - ✓XE Tivoli Warehouse
  - ✓Snapshot History
  - ✓Performance Warehouse
- DB2Plex Monitoring View
  - ✓CF structure & lock analysis
- Automation capabilities
- zIIP Engine utilization

OMEGAMON DB2 XE For DB2 PM/PE V4.1 Options & Interfaces

- **OMEGAMON XE GUI Interface**
  - **Real time and historical**
  - **Automation & alerts**
  - **Plex level information (CF, n-way)**

  OMEGAMON Classic
  - **3270 Interface command interface**
  - **Real Time & Historical**
- **OMEGAMON CUA**
  - **3270 interface**
  - **Real Time & Historical**
  - **Exception alerts**
- **PE GUI**
  - **GUI client interface**



OMEGAMON Analysis, Collection, And Tracing Options

- Real Time Analysis
  - Classic 3270, CUA 3270, PE GUI, Tivoli Enterprise Portal (TEP)
- Historical Analysis
  - Classic 3270 interface
    - Near Term Historical – last ‘n’ hours of history
    - Application Trace Facility – Performance trace for an interval
  - CUA 3270 interface
    - Near Term Historical – last ‘n’ hours of history
    - Application Trace Facility – Performance trace for an interval
  - Tivoli Enterprise Portal (TEP)
    - Tivoli Data Warehouse – snapshot history to the TDW
  - PE GUI interface
    - Snapshot history
    - Performance Warehouse - PWH

Relevant Information From The Classic Interface Main Menu (ZMENU)

```
 _________________ ZMENU    VTM     O2       V410./I DSNA 03/07/07 14:58:42   2
>  Help/News/Index PF1              Exit PF3                    PF Keys PF5
>    Type a selection letter at the left end of the top line and press ENTER
============================================================
>         OMEGAMON II FOR DB2 CLASSIC INTERFACE -- REALTI
_ S  SUMMARY ..............  Summary of DB2 activity
_ E  EXCEPTIONS ...........  Current or potential system p
_ T  THREAD ACTIVITY ......  Thread activity information
_ U  THREAD ACTIVITY ......  Thread activity information by Package
_ L  LOCKING CONFLICTS ....  Locking conflict information
_ R  RESOURCE MANAGERS ....  Resource manager, other DB2 subsystem information
_ A  APPLICATION TRACE ....  Trace and view application activity
_ D  DISTRIBUTED DATA .....  Distributed database system information
_ O  OBJECT ANALYSIS ......  Object and Volume
_ G  DB2 CONNECT SERVER ...  DB2 Connect/Gatewa
_ C  MVS CONSOLE ..........  MVS console to iss
_ B  DB2 CONSOLE ..........  DB2 console to iss
_ M  MISCELLANEOUS ........  Address space info
_ P  PROFILE ..............  Customize OMEGAMON
_ H  HISTORICAL ...........  Near-Term History
_ I  IFCID TRACE ..........  Start an IFCID Tra
_ V  SQL PA REPORTS .......  View SQL PA Report
_ Z  OTHER DB2 ............  Redirect monitorin
```

Menu option ‘R’ to access important subsystem information (Pool, storage, and SQL stats)

```
 _________________ ZRMMENU  VTM     O2       V410./I DSNA 03/07/07 15:02:26   2
>       Help PF1                                                  Back PF3
> R.

>                  Enter a selection letter on the top line.
================================================================================
>            RESOURCE MANAGERS AND OTHER DB2 SUBSYSTEM INFORMATION

_ A  BUFFER MANAGER .........  Buffer Manager Information
_ B  LOG MANAGER ............  DB2 Log Manager Information
_ C  EDM POOL ...............  EDM Pool Information
_ D  BIND STATISTICS ........  Bind Statistics
_ E  SUBSYSTEM MANAGER ......  DB2 Subsystem Support Manager Statistics
_ F  ACTIVE TRACES ..........  Current Trace Activity
_ G  START-UP OPTIONS........  IRLM and Stored Procedures Start-Up Options
_ H  DSNZPARM ...............  DB2 Installation Parameters
_ I  LOCK/CLAIM/DRAIN........  Lock Manager/Claim/Drain Statistics
_ J  SQL/RID POOL/PARALLEL...  SQL/RID Pool/Parallelism/Stored Proc. Information
_ K  OPEN/CLOSE STATISTICS...  Dataset Open and Close Statistics
_ L  DB2 COMMANDS ...........  DB2 Command Statistics
_ M  DB2 Storage  ...........  Storage Management Pool Summary
```

Menu option ‘H’ to access Near Term Historical Data

DB2 Statistics Information
Analyzing From The Subsystem Perspective

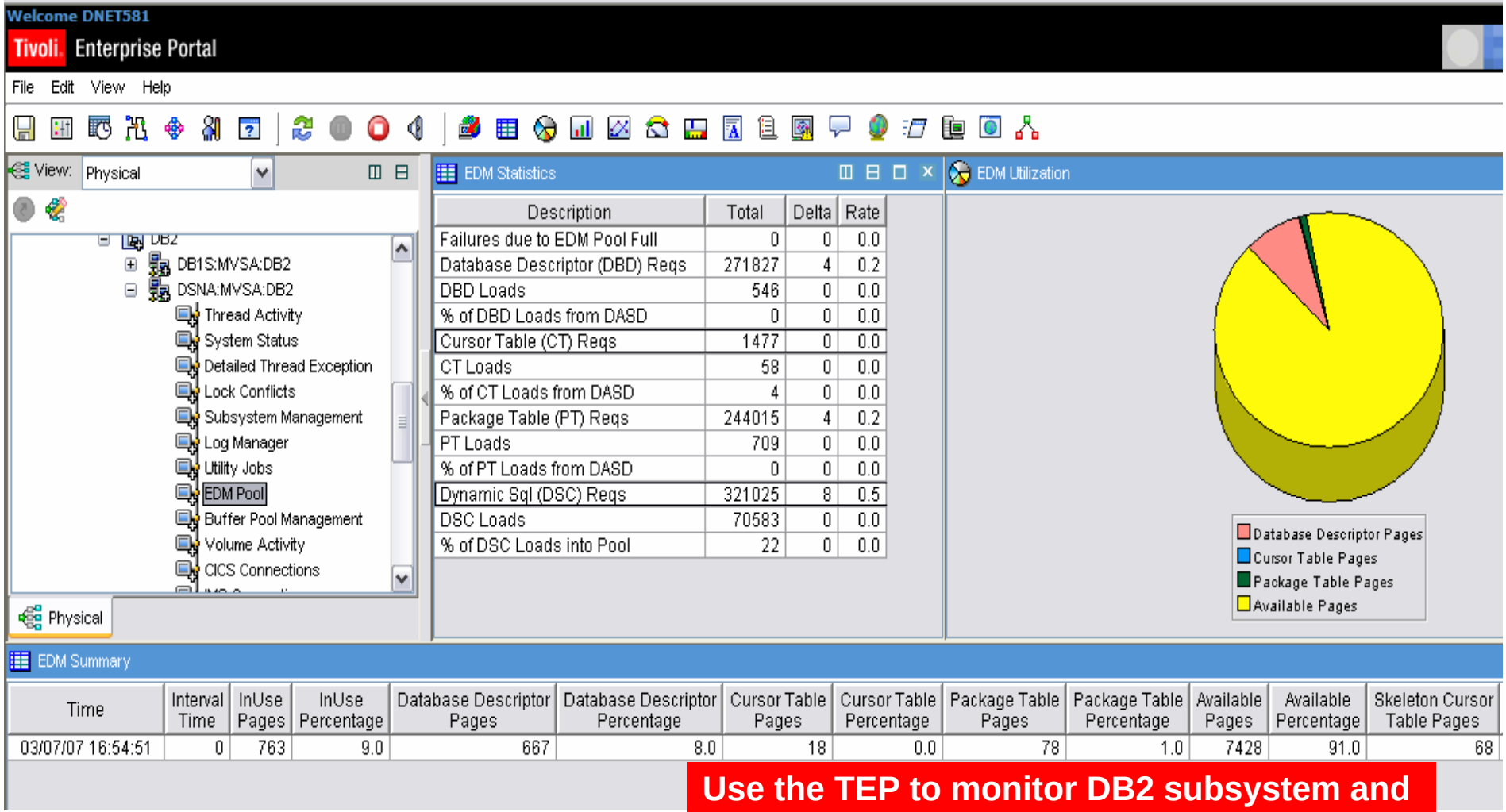
Prepare Statistics	INTERVAL QUANTITY
Copied from Cache	72465
No Match	5787
Implicit KEEPDYNAMIC(YES)	0
Avoided KEEPDYNAMIC(YES)	39556
Discarded - MAXKEEPD	0
Purged - DROP/ALTER/REVOKE	14

SQL Manipulative (DML)	TOTAL QUANTITY	INTERVAL QUANTITY
SELECT	577342	14
INSERT	350768	0
UPDATE	48286	1
DELETE	1010	0
OPEN CURSOR	987670	21
CLOSE CURSOR	936874	21
FETCH	169464K	35
PREPARE	671014	0

	TOTAL QUANTITY	INTERVAL QUANTITY
Dynamic Sql (DSC) Reqs	392268	0
DSC Loads	19589	0
% of DSC Loads into Pool	4.99%	.00%

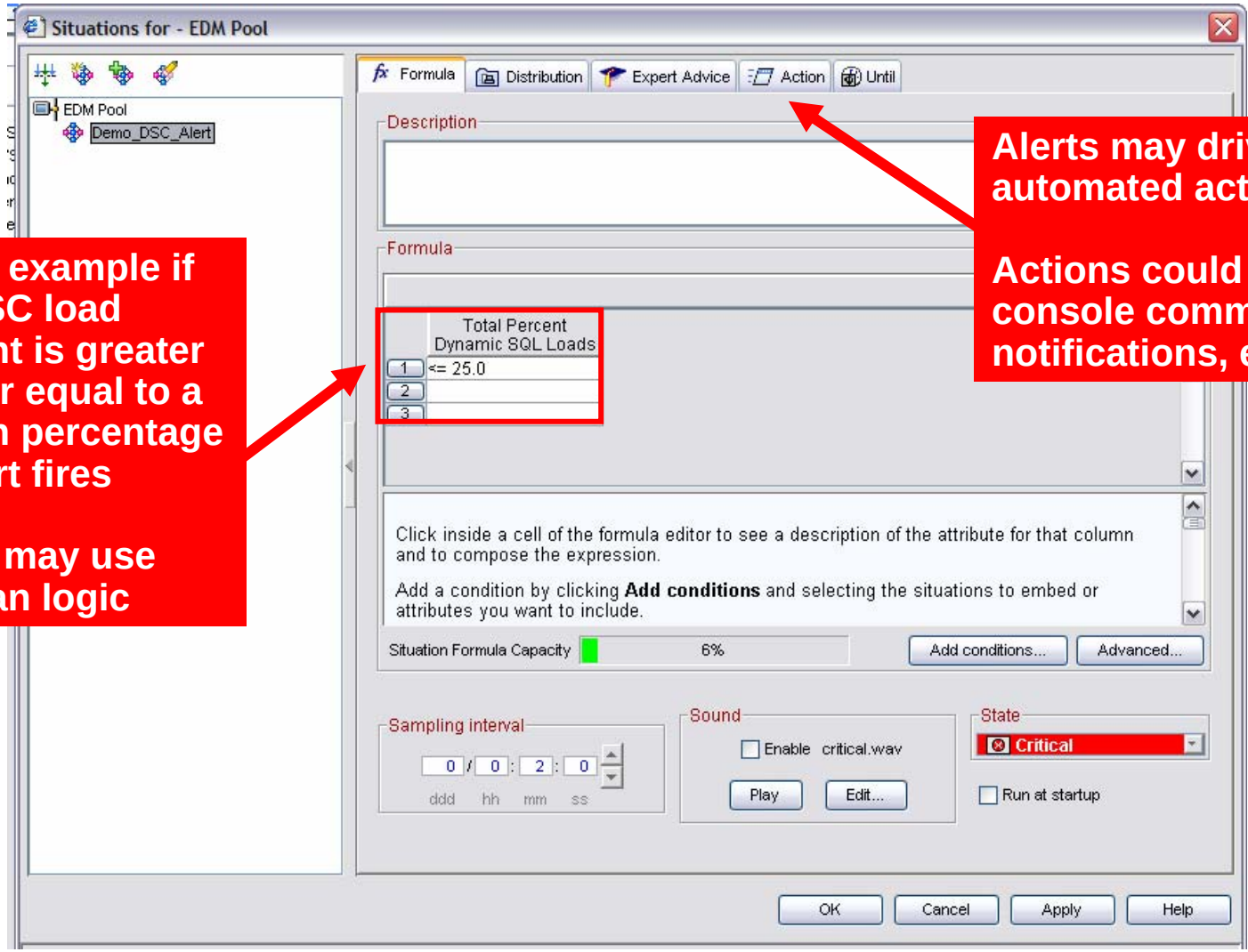
- OMEGAMON DB2 Statistics data shows the number and type of SQL calls performed for a given time interval
  - These counts are for the DB2 subsystem
- Select a desired time interval and generate the report
  - Note the number of prepares relative to other SQL related activity counts

OMEGAMON XE For DB2 PM/PE
Tivoli Enterprise Portal – TEP Interface



Use the TEP to monitor DB2 subsystem and application performance information

OMEGAMON Tivoli Enterprise Portal (TEP) Situations May Alert On Key Performance Metrics



In this example if the DSC load percent is greater than or equal to a certain percentage an alert fires

Alerts may use boolean logic

Alerts may drive automated actions

Actions could be console commands, notifications, etc.

OMEGAMON Provides Real Time, Snapshot, And Historical Data

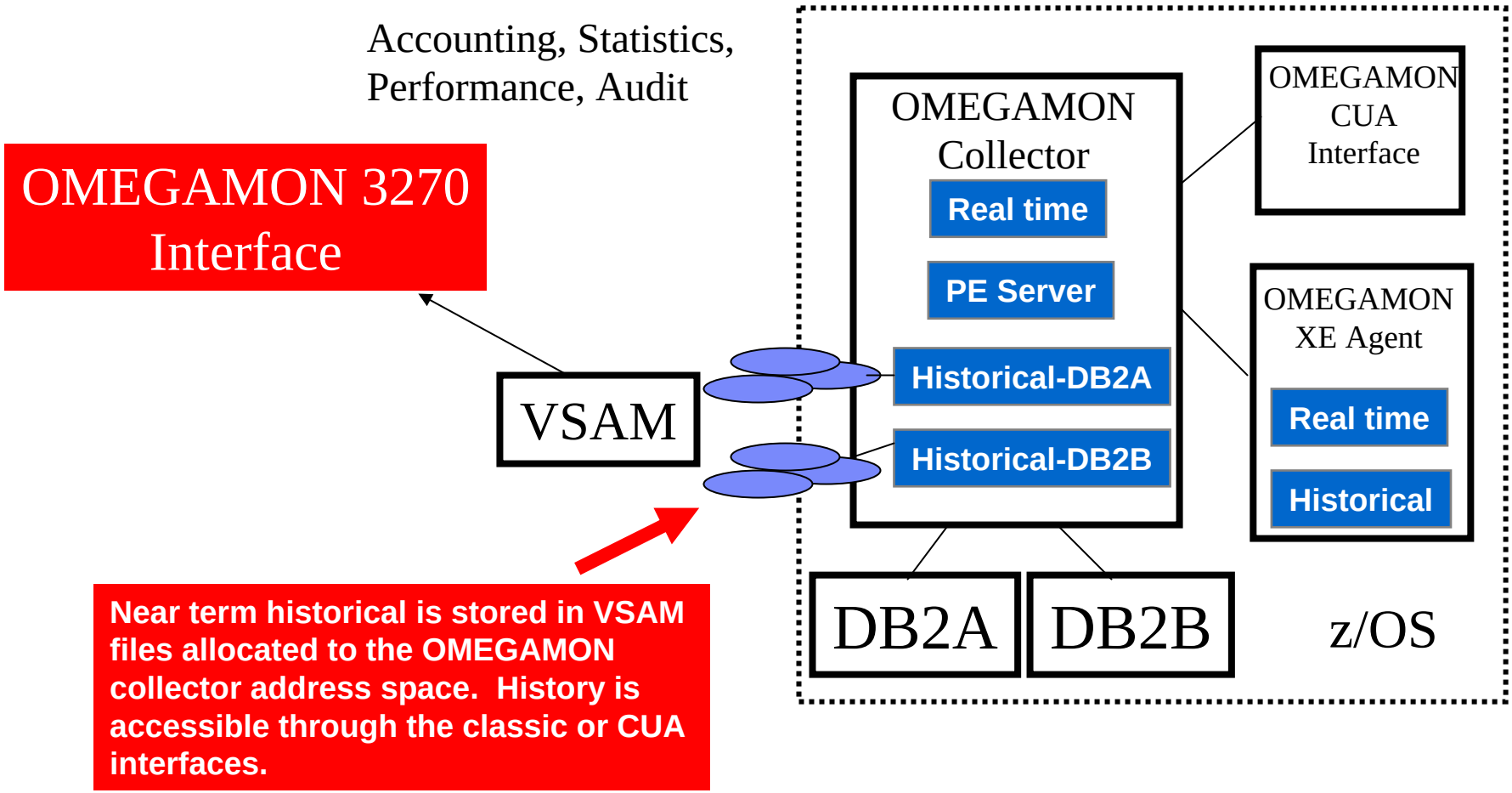
```
 _________________ ZEDMP    VTM     O2       V410./I DSNA 03/07/07 15:09:38   2
>       Help PF1          Back PF3            Up PF7          Down PF8
> A-EDM POOL SNAPSHOT                                          H-HISTORICAL
================================================================================
>                              EDM POOL INFORMATION
 EDMP
+ Collection Interval:  REALTIME                          S
+ Report Interval:       1 sec
+
+ Pool Usage    Pages  Pct                Percent of Total EDM Pool
+ ---------- ------- ---- |---10---20---30---40---50---60---70---80---90--100|
.....
+ STMT cache:             |
+   In Use      21217  19% |--------->    .    .    .    .    .    .    .    .|
+   Free        89378  81% |-------------------------------------->    .    .|
+   Total      110595 100% |---10---20---30---40---50---60---70---80---90--100|
+
+                                TOTAL  INTERVAL  /SECOND  /THREAD  /COMMIT
+                             QUANTITY  QUANTITY  (    1)  (    0)  (    9)
+                             --------  --------  -------  -------  -------
.....
+ Dynamic Sql (DSC) Reqs         91739         0      .00      .00      .00
+ DSC Loads                       7741         0      .00      .00      .00
+ % of DSC Loads into Pool        8.44%     .00%     n/c      n/c      n/c
```

Select option ‘A’ for pool snapshot data. Select option ‘H’ for Near Term History data

Portions of the EDMP major command relevant to the SQL cache

OMEGAMON XE For DB2 PM/PE

Near Term Historical Data Gathering And Usage

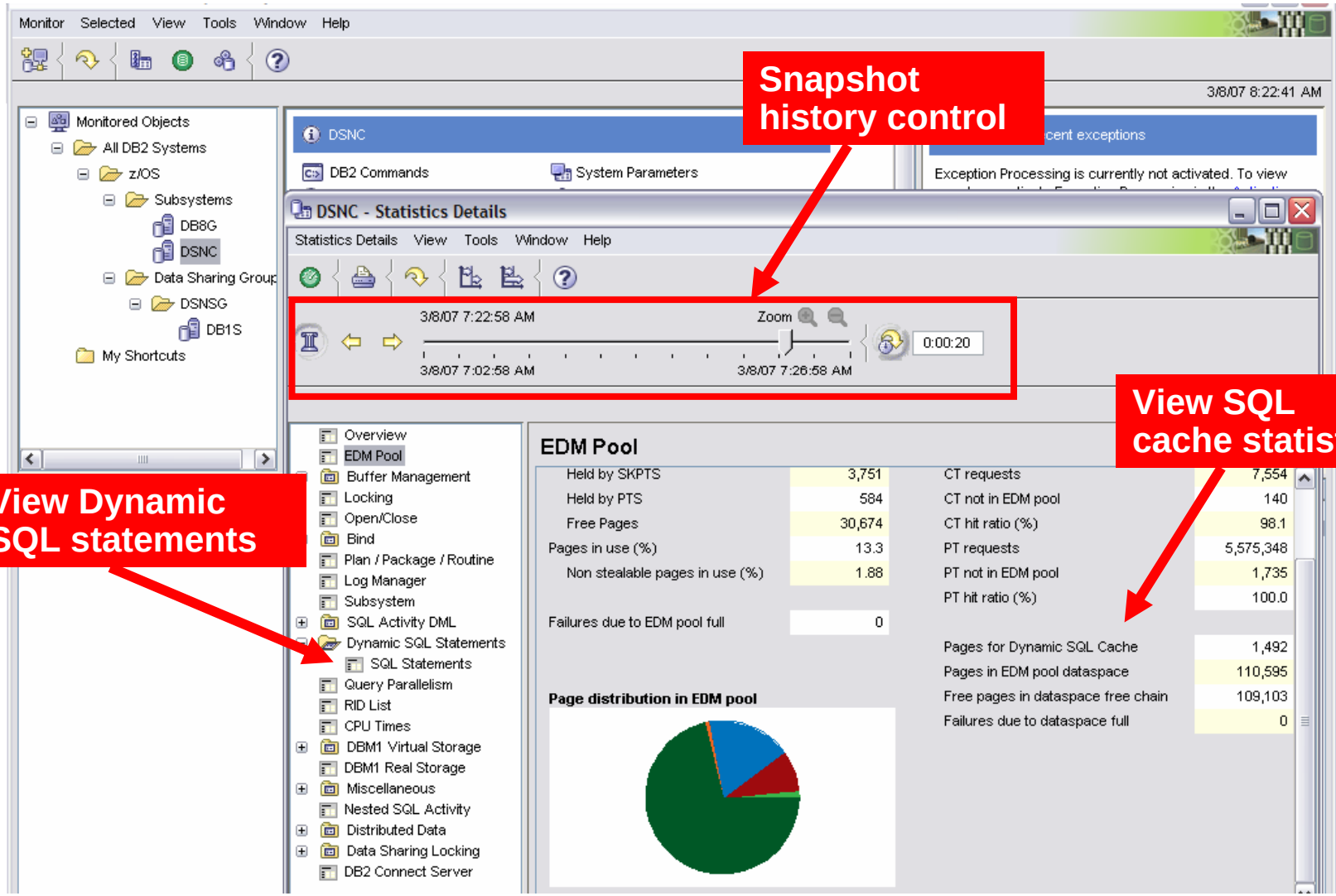


Near Term Historical Provides Ease Of Collection And Access – Most Recent 'N' Hours Of Data

```
 _________________ ZHEDS    VTM     O2       V410./I DSNA 03/07/07 15:14:52   2
>       Help PF1      Back PF3      Up PF7      Down PF8       Zoom PF11
> H.A.F
>                   Enter a selection letter on the top line.
>
> A-SUBSYSTEM SUPPORT      B-BIND              C-BUFFER POOL    D-GROUP BP
> E-DISTRIBUTED DATABASE   *-EDM POOL          G-LOG MANAGER    H-OPEN/CLOSE
> I-SQL/RID/PARALLEL/PROC  J-LOCK/CLAIM/DRAIN  K-GLOBAL LOCK    L-DB2 COMMANDS
> O-OPTIONS
================================================================================
>                  EDM POOL STATISTICS SUMMARY BY REPORT INTERV
 HEDS
+ Collection Interval:  15 min                              Start:
+ Report Interval:      15 min   Combine Level:  NONE       End:  03/07 15:14
+
+              Pages    DBD    DBD     CT     CT     PT     PT     DSC    DSC
+ Interval     in Use%  Pages  Load%  Pages  Load%  Pages  Load%  Pages  Load%
+ -----------  -------  -----  -----  -----  -----  -----  -----  -----  -----
+ 03/07 15:14       9%    667   .00%     18   .00%     78   .00%  10205  2.73%
+ 03/07 15:00       9%    667   .00%     18   .00%     78   .00%  10205  5.51%
+ 03/07 14:45       9%    667   .00%     16   .00%     78   .00%  10206  3.91%
+ 03/07 14:30       9%    667   .00%     16   .00%     78   .00%  10206  5.56%
+ 03/07 14:15       9%    667   .00%     16   .00%     78   .00%  10207  2.75%
```

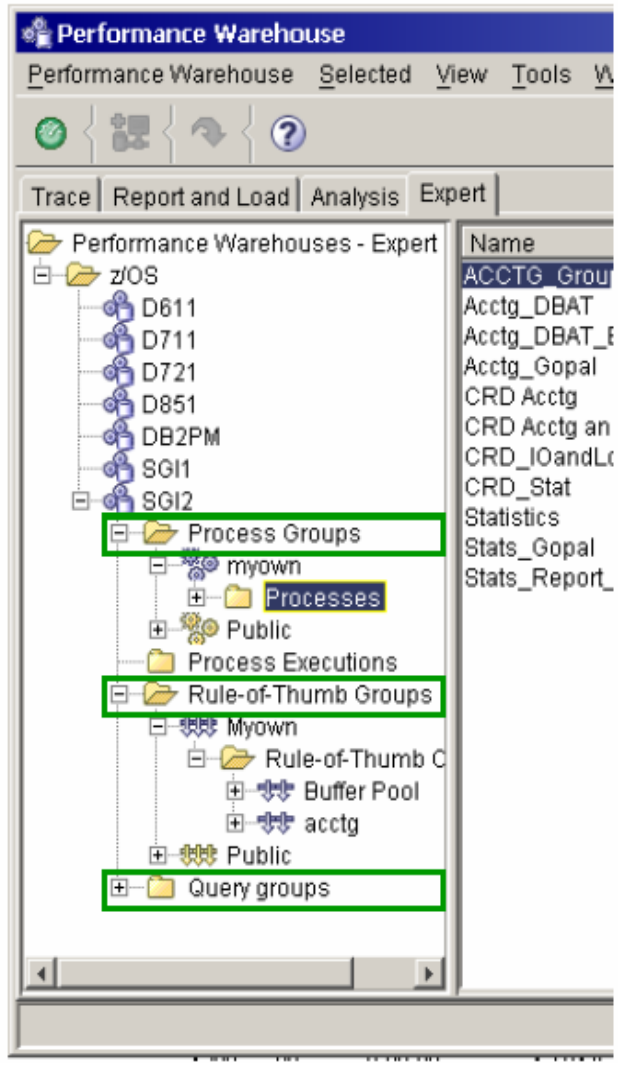
Select an interval and press F11 to drill down for time interval detail

PE GUI Provides Real Time Information And Snapshot History



Using PE GUI - Performance Warehouse – PWH Performance Analysis Collection, Retention, Reporting

- Infrastructure around the Performance Database tables
- The PE server component supports to control processes
  - Automatic creation and maintenance of the DB2 tables
    - Internal DB2 tables for process control
    - Performance DB2 tables for saving performance counters for subsequent analysis
  - To run PM Reports on the host configured and started from GUI with display of the report at the GUI
  - To build and schedule processes to collect, to prepare and to load DB2 performance data (DB2 event trace data) into the Performance Database
- Provides analysis support
  - Standard Rule of Thumb (ROT)
  - Standard SQL queries
  - Provides the capability to adapt and define customer own ROT and queries



Looking At DB2 Accounting Trace Information

- Review accounting data to understand what the applications are doing
  - # of SQL calls, type of SQL calls, duration of SQL In-DB2 activity,
  - DB2 SQL waits – I/O, lock/latch waits, and other waits
  - Stored Procedure activity, number of calls, SP scheduling delays
  - Thread level buffer stats
  - In-DB2 times, In-DB2 CPU times
  - Application level prepare and cache statistics
- Accounting traces are the starting point for performance analysis from the application perspective
  - Use Accounting data to isolate potential problem applications
- Look at number and type of prepares relative to overall SQL activity
  - Use counts to determine relative cost of dynamic SQL

PLAN Major Command Shows Thread Detail With Options To Show Relevant Detail

Select letter commands to see the relevant thread information

```
 _________________ ZTDTL    VTM     O2       V410./I DSNA 03/07/07 15:
> Help PF1
>       THREAD INFORMATION:  Enter a selection letter on the top line.
> *-THREAD DETAIL B-LOCK COUNTS C-LOCK WAITS     D-LOCKS OWNED  E-GLOBAL LOCKS
> F-CURRENT SQL   G-SQL COUNTS  H-DISTRIBUTED    I-BUFFER POOL  J-GROUP BP
> K-PACKAGES      L-RES LIMIT   M-PARALLEL TASKS N-UTILITY      O-OBJECTS
> P-CANCEL THREAD Q-DB2 CONSOLE R-DSN ACTIVITY   S-APPL TRACE   T-ENCLAVE
> U-LONG NAMES
================================================================================
>                                THREAD DETAIL
 PLAN
+ Thread:  Plan=DISTSERV  Connid=SERVER   Corrid=DB2JCC_APPLI Authid=DNET305
+ Dist   :  Type=DATABASE ACCESS, Luwid=G9A4AE78.G8EF.C0424ACA09F9=2272
+ Location  :  NDCDB201
 act
+ Thread Activity                        User Defined Functions
+ ------------------------------------   ------------------------------------
+ DB2 Status            =   WAIT-REMREQ   TCB Time (SQL)         = 00:00:00.000
+ MVS Status            =                 Wait for TCB Time      = 00:00:00.000
+ Total Elapsed Time    =  10:11:02.351   Elapsed Time           = 00:00:00.000
+ CP CPU Utilization    =         00.0%   Elapsed Time (SQL)     = 00:00:00.000
```

Examples Of Relevant DB2 Accounting Trace Data

```
+ Thread:  Plan=DSNESPRR  Connid=TSO      Corrid=DNET581      Authid=DNET581
+ Attach:  TSO            DB2=DSNC        MVS=MVSA
+ Time  :  Start=02/22/2007 21:04:12.116581     End=02/22/2007 21:04:15.186702
 act
+ Termination Status    =         DEALLOC     Commits              =          2
+ Total Elapsed Time    =    00:00:03.070     Aborts               =          0
+ Total CP CPU Time     =    00:00:00.021     Parallel Tasks       =          0
+ Total Stored Proc CPU =    00:00:00.000
+ Stored Proc Wait      =    00:00:00.000     Stored Proc Wait Cnt =          0
+
+ In-DB2 Times                                    Total
+ ------------------------------                  ------------
+ Elapsed Time                                    00:00:00.018
+ CP CPU Time                                     00:00:00.012
+ Stored Procedure CPU Time                       00:00:00.000
```

In-DB2 CPU time for the application

```
+ Thread:  Plan=DSNESPRR  Connid=TSO      Corrid=DNET581      Authid=DNET581
+ Attach:  TSO            DB2=DSNC        MVS=MVSA
+ Time  :  Start=02/22/2007 21:04:12.116581     End=02/22/2007 21:04:15.186702
 sqls
+ Commit         =       2  Abort          =       0  Select         =       0
+ Open Cursor    =       1  Close Cursor   =       1  Fetch          =     250
+ Insert         =       0  Delete         =       0  Update         =       0
+ Describe       =       0  Lock Table     =       0  Prepare        =       1
+ Grant          =       0  Revoke         =       0  Set Rules      =       0
+ Increm Bind    =       0  Label/Comm On  =       0  Set SQLID      =       0
+ Set Host Var   =       0  Set Connection =
+ Connect Type 1 =       0  Connect Type 2 =
+ Rename Table   =       0  Hold Locator   =
+ Release        =       0  Assoc Locator  =
```

Counts of SQL calls including prepares

Counts of cache hits and misses

```
+ Prepare Statistics:
+  Copied from Cache      =       0    Implicit - KEEPDYNAMIC(YES)    =       0
+  No Match               =       1    Avoided - KEEPDYNAMIC(YES)     =       0
+  Discarded - MAXKEEPD   =       0    Purged - DROP/ALTER/REVOKE     =       0
+
```

- Use accounting data to determine the impact of dynamic SQL on the application time line

Near Term History Collects Accounting And Optionally Dynamic SQL Call Information

```
 _________________ ZHAGPL   VTM      O2       V410./I DSNA 03/07/07 16:03:17   3
> Help PF1          Back PF3          Up PF7          Down PF8          Zoom PF11
>
>                    Enter a selection letter on the top line.
>
> *-BY PLAN       B-BY AUTHID       C-BY PLAN,AUTHID       D-BY AUTHID,PLAN
> O-OPTIONS
=============================================================
>                             THREAD HISTORY BY PLAN
 HAGP
+ Report Interval:    15 mins                        Start:
+ Report Filtered:       NO                           End:
 plan
+
+                                      DLk/   In-DB2   In-DB2
+ Plan      Thrds Commit Abrt   DML   TOut   Elap Tm CPU Tm
+ -------- ----- ------ ---- ----- ----   ------- -------
+ DISTSERV    37     37    0     82    0        .2      .04
+ DEMOTHD      1      0    3     37    0        .0      .00
=============================================================
```

F11 zoom to see thread detail

```
 _________________ ZHTCALL  VTM      O2       V410./I DSNC 02/13/07  8:45:55   2
> Help PF1               Back PF3               Up PF7                  Down PF8
>          THREAD HISTORY:  Enter a selection letter on the top line.
> A-THREAD DETAIL  B-LOCK COUNTS  C-LOCK WAITS     D-GLOBAL LOCKS  E-SORT/SCAN
> *-DYNAMIC SQL    G-SQL COUNTS   H-DISTRIBUTED    I-BUFFER POOL   J-GROUP BP
> K-PACKAGE SUMMARY L-RES LIMIT   M-PARALLEL TASKS N-SQL PA
================================================================================
>                      THREAD HISTORY DYNAMIC SQL CALLS
 HPLN
+ Thread:  Plan=DEMOTHD  Connid=RRSAF    Corrid=DEMO.ADMI Authid=DEMOID
+ Attach:  RRSAF          DB2=DSNC        MVS=MVSA
+ Time  :  Start=02/13/2007 07:18:55.917137     End=02/13/2007 07:43:55.290594
 call
+
: Select Call=NEXT      (FIRST/LAST/NEXT/PREV/+nnnnn/-nnnnn/Snnnnn)
+
+                              SQL Statement      (       2 of       5)
+
+ SELECT COUNT(*)
+    FROM “DEMO".IBMQREP_SENDQUEUES
+    WHERE STATE_INFO LIKE '%ASN____E%'
```

SQL text stored in NTH

Near Term History Collection Options

```
 _________________ ZH2IN    VTM     O2       V410./I DSNA 03/07/07 16:10:47   2
>     Help PF1         Back PF3                Up PF7          Down PF8
> H.C.A
>    NEAR-TERM HISTORY INFORMATION: Enter a selection letter on the top line.

> *-COLLECTION OPTIONS        B-RECORD INFORMATION           C-DATASET STATUS
================================================================================
=
>                  NEAR-TERM HISTORY DATA COLLECTION OPTIONS
 COPT
+                              H2 Collection Options
+
+ DB2sys       = DSNA          Writeoption   = VSAM       Interval    =   15
+ Archivejcl   = ARCVDSNA      Tracebufsz    = 300K       Ifireadtime = 010000
+ Maxhours     = 24            Suspcoll      = Yes        PostPCT     = 70
+ Destination  = None
+
+ Statistics   = Yes           Dsnzparm      = Yes
+ Auditing     = (1 2 3 4 5 6 7 8 )
+ Accounting   = (1 2 3 7 8 ) Sort          = Yes        Lock Contention = Yes
+                             Scan          = Yes        Lock Suspension = Yes
+                             Dynamic SQL   = Yes        Negative SQL    = Yes
+
+ H2 Data Sets :
+                DEMO.DEMOMVS.DSNA.RKD2VS01
```

Dynamic option enables collection of dynamic SQL text in NTH

Problem Isolation Using Performance Traces The Most Detailed Level Of DB2 Tracing

- Accounting traces can be used to isolate performance issues down to the plan/dbrm/package level
- With dynamic SQL applications there may be more permutations and combinations of SQL
  - Tracing may be needed to capture SQL call information for analysis
- Use Performance traces to isolate down to the SQL statement level and view detail activity within the statement level
- Use performance traces judiciously
  - Concerns include:
    - Trace overhead
    - Quantity of data generated
    - Retention and post-processing of the data

Application Trace Facility - ATF

```
 _________________ ZATRQ    VTM     O2       V410./I DSNA 03/07/07 16:23:21   2
>         Help PF1                                              Back PF3
> *-SPECIFY TRACE    B-VIEW TRACE       C-STOP TRACE       D-SELECT DSN
> E-VIEW DATASET     F-STOP VIEW        G-CREATE VSAM LDS
================================================================================
>                          SPECIFY APPLICATION TRACE
 ATRQ
+ Type DB2 Plan name to be traced. Also, provide additional optional
+ selection information to limit trace output. To save trace records
+ for later viewing you must specify a data set name for DSN
+
:            DSN= ____________________________________________ Data set name
:           TIME= 005      Number of mins to trace (001-060)
:       PLANNAME= DSNESPRR Plan name or ALL for all active threads
:         AUTHID= ________ DB2 authorization identifier
:        TSOUSER= ________ TSO USERID (TSO foreground app)
:        JOBNAME= ________ Jobname (TSO batch app)
:       CICSTRAN= ____     CICS trans id
:       CICSCONN= ________ CICS connection id)
:        PSBNAME= ________ IMS PSB name
:          IMSID= ________ IMS ID of the IMS region
:       LOCKDATA= Y        Collect DB2 lock trace recs? (Y/N)
:       SCANDATA= Y        Collect DB2 scan trace recs? (Y/N)
:        SQLDATA= Y     Collect DB2 sql  trace recs? (Y/N)
:       THRDDATA= Y     Collect DB2 thread trace recs?  (Y/N)
:       CONNDATA= Y     Collect DB2 connect trace recs? (Y/N)
:            SMF= N     Write trace data to SMF? (Y/N)
:            GTF= N     Write trace data to GTF? (Y/N)
:        MEMSIZE= 02    Collection workarea memory size (01-04 meg)
```

ATF allows for performance tracing for a specified interval

Collection is to a VSAM file or OMEGAMON memory

Trace analysis may be done interactively within OMEGAMON

Using The Application Trace Facility

```
 _______________ ZATRQ    VTM     O2       V410./I DSNA 03/07/07 16:23:21   2
>         Help PF1                                          Back PF3
> *-SPECIFY TRACE    B-VIEW TRACE       C-STOP TRACE       D-SELECT DSN
> E-VIEW DATASET     F-STOP VIEW        G-CREATE VSAM LDS
================================================================================
>                         SPECIFY APPLICATION TRACE
 ATRQ
+ Type DB2 Plan name to be traced. Also, provide additional optional
+ selection information to limit trace output. To save trace records
+ for later viewing you must specify a data set name for DSN
+
:            DSN= ____________________________________________ Data set name
:           TIME= 005      Number of mins to trace (001-060)
:       PLANNAME= DSNESPRR Plan name or ALL for all active threads
:         AUTHID= DNET581_ DB2 authorization identifier
:        TSOUSER= ________ TSO USERID (TSO foreground app)
:        JOBNAME= ________ Jobname (TSO batch app)
:       CICSTRAN= ____     CICS trans id
:       CICSCONN= ________ CICS connection id)
:        PSBNAME= ________ IMS PSB name
:          IMSID= ________ IMS ID of the IMS region
:       LOCKDATA= Y        Collect DB2 lock trace recs? (Y/N)
:       SCANDATA= Y        Collect DB2 scan trace recs? (Y/N)
:        SQLDATA= Y     Collect DB2 sql  trace recs? (Y/N)
:       THRDDATA= Y     Collect DB2 thread trace recs?  (Y/N)
:       CONNDATA= Y     Collect DB2 connect trace recs? (Y/N)
:            SMF= N     Write trace data to SMF? (Y/N)
:            GTF= N     Write trace data to GTF? (Y/N)
:        MEMSIZE= 02    Collection workarea memory size (01-04 meg)
```

- Performance traces allow for analysis to the SQL call
- Provides granularity to see SQL call level detail and see the impact of dynamic SQL on the application
- ATF provides an SQL index overview

```
+ Planname=DSNESPRR   Connid=TSO         Corrid=DNET581        Authid=DNET581
+
+ Call Type         Stm#  Program  Count InDB2 Time    MRet Rws Pc Rws DM Rws RD
+ ---------------- ----- -------- ----- ------------ ---- ------ ------ ------
+ PREPARE            116 DSNESM68     1 00:00.00251     0      3      1      0
+ OPEN CURSOR        190 DSNESM68     1 00:00.00005     0      0      0      0
+ FETCH              183 DSNESM68   250 00:00.00694     0      0      0      0
+ CLOSE CURSOR       197 DSNESM68     1 00:00.00002     0    499    249    250
================================================================================
```

A Performance Trace Example
Understanding The Cost Of A Full Prepare

```
>                          APPLICATION TRACE SQL DETAIL
 ATD1
+ Planname=DSNESPRR   Connid=TSO        Corrid=ADCDA          Authid=ADCDA
+
: Control=NEXT    (Valid options are FIRST/LAST/NEXT/PREV/+nnnnn/-nnnnn/Snnnnn)
+ Current=000001  Total Number of SQL Calls=000253
+
+ Start Time    Progname  SQL Call         Stmt# Retcode  InDB2 Time    InDB2 CPU
+ ------------ --------  ---------------- ----- -------  ------------ --------
+ 21:14:24.252 DSNESM68  PREPARE          00000       0  00:02.39784      .05615
+
+ Data Rows    Rows    Rows     Rows     Rows    Rows    Rows    Rows    Pages  Pages
+ Type Proces Looked Qual/DM Qual/RD Update Insert Delete De/Ref Scand  Sc/Ref
+ ---- ------ ------ ------- ------- ------ ------ ------ ------ ------ ------
+ INDX      6      2       0       0      0      0      0      0     10      0
+ DATA      1      1       0       0      0      0      0      0      1      0
```

Example
CPU cost of a full Prepare

***Note – Measure on your system, mileage will vary**

Note scan activity on Catalog for APS information

- Use performance trace data to analyze and understand detailed SQL call activity
- In this example tracing may be used to assess the cost of a prepare
  - This may be used for later analysis

The Cost Of A Short Prepare

Same Statement - Big Difference In The Prepare CPU Cost

```
>                          APPLICATION TRACE SQL DETAIL
 ATD1
+ Planname=DSNESPRR   Connid=TSO        Corrid=ADCDA          Authid=ADCDA
+
: Control=NEXT    (Valid options are FIRST/LAST/NEXT/PREV/+nnnnn/-nnnnn/Snnnnn)
+ Current=000001  Total Number of SQL Calls=000253
+
+ Start Time   Progname  SQL Call         Stmt# Retcode  InDB2 Time   InDB2 CPU
+ ------------ --------  ---------------- ----- -------  ------------ ---------
+ 21:16:57.144 DSNESM68  PREPARE          59739       0  00:00.00094     .00085
+
+ Data Rows    Rows    Rows     Rows     Rows    Rows    Rows    Rows    Pages  Pages
+ Type Proces Looked Qual/DM Qual/RD Update Insert Delete De/Ref Scand  Sc/Ref
+ ---- ------ ------ ------- ------- ------ ------ ------ ------ ------ ------
+               (No Data Activity Located For This Call)
+
```

Example CPU cost of a short Prepare

***Note – Measure on your system, mileage will vary**

Note no scan activity on Catalog for the short Prepare

- In this example the full Prepare is 66 times more expensive than the short Prepare
- Trace and analyze on your system to understand the cost of Prepare

Additional Analysis And Collection Options
DSC Cache Snapshot Analysis

- Useful to trace and analyze activity in the EDM SQL cache
- Provides a mechanism to view dynamic SQL activity in the SQL cache with statement level information in a lower overhead manner
- Shows SQL level counts and execution times
- Position cursor and F11 zoom to see SQL detail

```
_______________ ZEDDT    VTM     O2       V410./I DSNA 02/13/07  8:39:04   2
>  Help PF1    Back PF3     Up PF7     Down PF8     Sort PF10     Zoom PF11
>
>      A=DYNAMIC SQL CACHE BY AUTHID                      B=*
================================================================================
>                    DYNAMIC SQL CACHE STATISTICS
>
>   place an 'a' before the EDDT command to get average for times and counts
>
 EDDT
+ *
+  Times      CPU           Elapsed        Wait        Get-    Sync    Sync
+  Exec.      Time           Time          Time        Pages   Reads  Writes
+ ------  ------------  ------------  ------------  ------  ------  ------
+  12093  00:00:00.965  00:00:01.224  00:00:00.000   12093       0       0
+  12093  00:00:00.722  00:00:01.244  00:00:00.000   12093       0       0
+   8062  00:00:00.482  00:00:00.545  00:00:00.000    8062       0       0
+   8062  00:00:01.437  00:00:02.497  00:00:00.045       0       5       0
+   4031  00:00:00.211  00:00:00.357  00:00:00.000    4031       0       0
+   4031  00:00:00.472  00:00:00.799  00:00:00.065       0       5       0
+   2421  00:00:00.238  00:00:00.400  00:00:00.124    4844      16       0
+   2017  00:00:00.233  00:00:00.264  00:00:00.000    7670       0       0
+   1793  00:00:00.359  00:00:00.557  00:00:00.000    5379       0       0
+   1793  00:00:00.353  00:00:00.508  00:00:00.001    3586       1       0
```

Use DSC Cache Analysis To See SQL Text Detail

```
 _________________ ZEDD3    VTM     O2       V410./I DSNA 02/13/07  8:37:48   2
>  Help PF1        Back PF3         Up PF7         Down PF8
> A-SQL PA
================================================================================
>                EDM SNAPSHOT DYNAMIC SQL CACHE STATISTICS
>     statistics require that monitor class 1 and ifcid 318 be started
 EDD3
+             Authorization Id: DEMOID
+
+  UPDATE “DEMO".IBMSNAP_REGISTER SET SYNCHTIME = ? WHERE DEMO _RECORD =
+   'Y' AND SYNCHTIME < ?
+
+  Times Executed                1791   Synchronous Buffer Reads          0
+  Getpages                      5373   Rows Examined                     0
+  Rows Processed                1791   Sorts Performed                   0
+  Index Scans                   1791   Tablespace Scans                  0
+  Parallel Groups Created          0   Synchronous Writes                0
+  Elapsed Time           00:00:00.556   CPU Time               00:00:00.358
+  Wait for Synch I/O     00:00:00.000   Wait for Lock/Latch    00:00:00.000
+  Synch Exec Switch      00:00:00.000   Wait for Global Locks  00:00:00.000
+  Wait Othr Thread Read  00:00:00.000   Wait Othr Thread Write 00:00:00.000
+  Isolation Bind                  UR   Currentdata Bind                  N
+  Dynamic rules Bind               R   Current Degree                    1
+  Current Rules                    D   Current Precision                 N
```

EDM snapshot shows SQL text and performance information

SQL And Thread Options Impact DB2 Storage Utilization OMEGAMON Provides Storage Utilization Information

```
+ Total variable storage (MB)                      = 104.902
+   Total agent local storage (MB)                 = 60.906
+     Total agent system storage (MB)              = 29.664
+       Prefetch engines                           = 18
+       Deferred write engines                     = 250
+       Castout engines                            = 0
+       GBP write engines                          = 0
+       P-Lock/notify exit engines                 = 0
+   RID pool storage (MB)                          = .348
+   Pipe manager sub pool storage (MB)             = .941
+   Local dynamic stmt cache cntl blks (MB)        = 5.891
+   Thread copies of cached sql statements (MB)    = 15.703
+     In use storage (MB)                          = .296
+     Statements count                             = 40
+     High water mark for allocated stmts (MB)     = .296
+       Statement count at high water mark         = 40
+       Date at high water mark                    = 2007-02-22 21:48:37
```

- OMEGAMON IFCID 225 storage displays provides storage utilization information in the DBM1 address space
- When determining an optimization strategy for dynamic SQL these numbers should be reviewed to understand the impact of the settings on overall DB2 storage usage

Forging An OMEGAMON DB2 Dynamic SQL Analysis Strategy

- Gather data, measure and establish a baseline
  - Gather Statistics data to determine
    - The number of prepares, types of prepares, cache activity counts, the relative amount of dynamic SQL on the subsystem
  - Gather Accounting data for key applications to determine
    - The number of prepares, types of prepares, cache activity counts
  - Use the data to establish a baseline starting point for analysis
- Use the Logical Tuning Methodology
  - Try calculating the relative cost of dynamic SQL
    - The cost as reflected on the subsystem
    - The cost as reflected on critical applications
  - Understand the application time line for key applications
  - Ask the question
    - “Is dynamic SQL an issue?” – If yes, how much?

Forging A Dynamic SQL Analysis Strategy - continued

- Determine a trace collection and retention strategy
  - Statistics traces – ongoing – SMF, NTH, Snapshot, PWH
  - Accounting traces – ongoing - SMF, NTH, Snapshot, PWH
  - Performance traces – determine based upon analysis needs versus cost of collection, retention, and analysis
- Exploit the facilities provided by OMEGAMON XE For DB2 PM/PE
  - Classic Interface
    - Thread detail, subsystem detail, Near Term Historical, Application Trace Facility (ATF)
  - PE GUI
    - Thread detail, subsystem detail, snapshot history, Performance Warehouse (PWH)
  - Tivoli Enterprise Portal (TEP)
    - High level analysis, alerting, automation
    - Tivoli Data Warehouse history

Additional Monitoring Tools And Options
DB2 QUERY MONITOR
SQL Monitor

- Static / Dynamic SQL
- Monitoring Profile – determine what to capture
- Exception processing
- History
- Auxiliary Functions
  - Capture negative return codes
  - Capture DB2 commands
  - Host Variables

✓ISPF and GUI interfaces
✓Supports DB2 z/OS
✓Exploits DB2 V9
✓Interfaces with
- DB2 SQL Performance Analyzer
- OMEGAMON

DB2 SQL PERFORMANCE ANALYZER
Enhanced Explain

- Forecasts SQL performance
  - Response times
  - CPU times
  - I/O counts
  - Cost of query
- Reports
- Preemptive governor
- Easy Explain
- What If Analysis

✓ISPF and batch interfaces
✓Supports DB2 z/OS
✓Exploits DB2 V8
✓Integrates with
- DB2 Query Monitor
- DB2 Path Checker
- OMEGAMON

SQL Performance Analyzer Integration With OMEGAMON Classic Interface

```
 _________________ ZSQL     VTM     O2       V410./I DSNC 03/08/07  7:54:41   2
> Help PF1                                                      Back PF3

>       THREAD INFORMATION:  Enter a selection letter on the top line.

> A-THREAD DETAIL B-LOCK COUNTS C-LOCK WAITS     D-LOCKS OWNED  E-GLOBAL LOCKS
> *-CURRENT SQL   G-SQL COUNTS  H-DISTRIBUTED    I-BUFFER POOL  J-GROUP BP
> K-PACKAGES      L-RES LIMIT   M-PARALLEL TASKS N-UTILITY      O-OBJECTS
> P-CANCEL THREAD Q-DB2 CONSOLE R-DSN ACTIVITY   S-APPL TRACE    T-ENCLAVE
> U-LONG NAMES    V-SQL PA
================================================================================
>                           SQL CALL BEING EXECUTED
 PLAN
+ Thread:  Plan=DSNESPRR  Connid=SERVER   Corrid=JAVAW.EXE    Authid=DEMO
+ Dist   :  Type=DATABASE ACCESS, Luwid=G941FEF1.BB05.070307143745=40870
 call
+      SQL call is active, call information is as follows :
```

Issue command from classic interface to execute SQL PA

```
 _________________ ZSQPO00  VTM     O2       V410./I DSNC 03/08/07  7:52:49   2
> Help PF1          Back PF3            Up PF7              Down PF8
>
>                   SQL PA Analysis: EXPLAIN Output
>
> *-EXPLAIN    B-QLIMIT    C-QTRACE    D-SYSPRINT  E-ANLSQL    F-JOBERR
>
================================================================================
 SQPO
> Report=000113 Plan=DSNESPRR Package=**NONE** Date=2006-09-21 Time=09.50.40
09:50:43.053               SQL Performance Analyzer               Version  3.1.
 09-21-2006                Enhanced Explain Report                Level  3N-310

                             SQL PA Parameters
(ANLPARM)
   REPORTS ALL
   SHOWALT YES
   DELIMIT NONE
   VIADRDA +OFF+
   USEPLAN +OFF+
```

Summary

- Dynamic SQL is being used more and more pervasively in many applications
- Dynamic SQL poses its own unique performance considerations and challenges
- Take advantage of the facilities of OMEGAMON to monitor, manage, and tune dynamic SQL