

Monitoring and Reporting

Monitoring	2-2
Traces	2-2
DB2 traces	2-3
Diagnostic traces for the attachment facility	2-4
Diagnostic traces for the IRLM	2-5
New traces for DB2 V7	2-6
Real Time Statistics	2-9
Starting the RTS database	2-14
DB2 Performance Monitor	2-16
Query Monitor	2-19
What else to monitor?	2-21
Planning for performance	2-23
SQL/PA	2-24
DB2 Estimator	2-24

Monitoring

There is no shortage of IBM monitoring and reporting tools for a DB2 system.

- MVS Resource Measurement Facility
- IMS, CICS, DB2 Monitor components
- Trace records
- GTF traces
- DB2 Performance Monitor (DB2PM)
- SQL Performance Analyzer
- Query Monitor
- DB2 Estimator
- DB2 Tabular Reports
- System Oriented Reports
- Workload Oriented Reports
- DB2PM Graphic Reports
- Service Level Reporter output

You can also use 3rd party products such as:

- Omegamon
- Subsystem analyser
- Detector (Platinum)

Traces

You can use the following traces for problem determination:

- DB2 trace
- IMS attachment facility trace
- CICS trace
- Three TSO attachment facility traces
- CAF trace stream
- OS/390 RRS trace stream
- MVS component trace used for IRLM

DB2 Traces

DB2 traces record the following types of data:

Statistics	allows you to perform DB2 capacity planning and to tune the entire set of DB2 programs.
Accounting	allows you to assign DB2 costs to individual authorization IDs and to tune individual programs.
Performance	subsystem events, which can be used to do program, resource, user, and subsystem-related tuning.
Audit	monitor DB2 security and access to data.
Monitor	data used by DB2 monitor application programs.

To use the trace commands you will need one of the following types of authority:

- SYSADM or SYSOPR authority
- Authorization to issue start and stop trace commands (the TRACE privilege)
- Authorization to issue the display trace command (the DISPLAY privilege).

The trace commands include:

START TRACE	Invokes one or more different types of trace.
DISPLAY TRACE	Displays the current trace options.
STOP TRACE	Stops any trace.
MODIFY TRACE	Changes the trace events (IFCIDs) being traced for a specified active trace.

There are several parameters which can further qualify the scope of a trace. You can trace specific events within a trace type as well as events within specific DB2 plans, authorization IDs, resource manager IDs and location.

You can also specify the destination to which trace data will be sent.

When you install DB2, you can specify which trace types and classes will start automatically when DB2 starts.

Diagnostic traces for the attachment facilities

- **IMS** provides a trace facility which shows the flow of requests across the connections from the control and dependent regions to DB2. This trace is recorded on the IMS log if the appropriate options are specified. You can print it with DFSERA10 plus a formatting exit module.

Also, the IMS attachment facility of DB2 provides an internal wrap-around trace table that is always active. When certain unusual error conditions occur, these trace entries are externalized on the IMS log.

- You can use CETR to control the **CICS** trace facility. CETR gives you a series of menus that you can use to set CICS trace options.
- The **TSO** attachment facility provides three tracing mechanisms:
 - The DSN trace stream
 - The CLIST trace facility
 - The SPUFI trace stream
- The **call attachment facility** trace stream uses the same ddname as the TSO DSN trace stream, but is independent of TSO.
- The **RRSAF** trace stream also uses the same ddname as the TSO DSN trace stream, but is independent of TSO. An RRSF internal trace will be included in any ABEND dump produced by RRSF. The trace provides a history of RRSF usage that can help you when diagnosing errors in RRSF.

Diagnostic trace for the IRLM

These MVS commands control the diagnostic traces for the IRLM:

MODIFY irlmproc,SET,TRACE

Sets dynamically the maximum number of trace buffers for each trace type. This value is used only when the external component trace writer is not activated.

MODIFY irlmproc,STATUS,TRACE

Displays the status of traces and the number of trace buffers used for each trace type. Also displays whether or not the external component trace writer is active for the trace.

START irlmproc,TRACE=YES

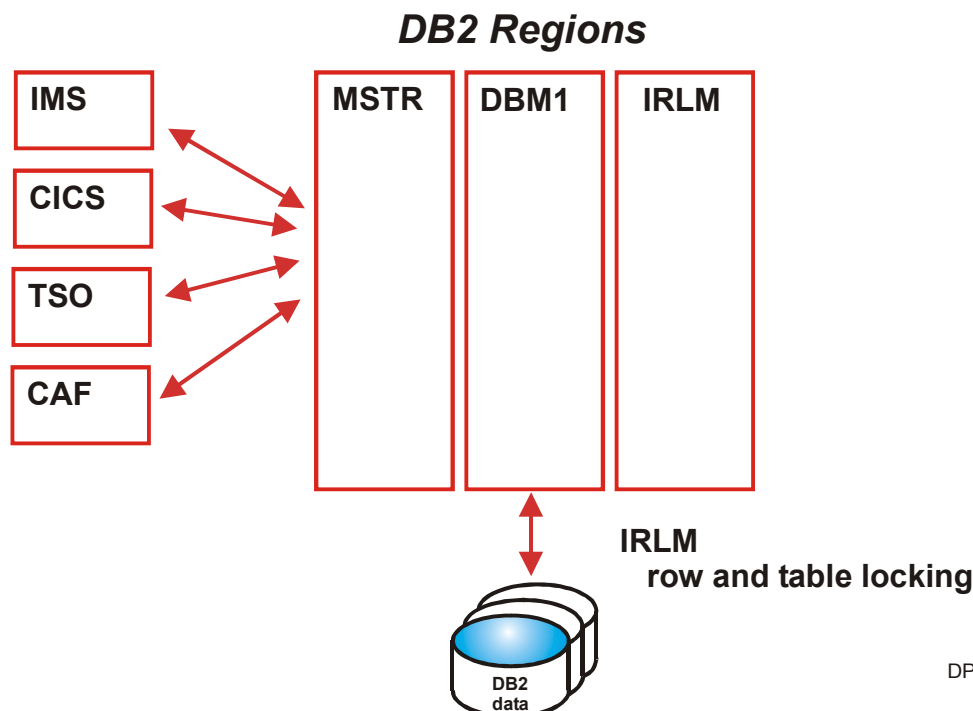
Captures traces in wrap-around IRLM buffers at IRLM startup.

TRACE CT

Starts, stops, or modifies a diagnostic trace for IRLM. The TRACE CT command does not know about traces that are started automatically during IRLM startup.

Recommendations:

- Do not use the external component trace writer to write traces to the data set.
- Activate all traces during IRLM startup.
Use the command `START irlmproc,TRACE=YES` to activate all traces.



DPA-200

New traces for DB2 V7

Two IFCIDs have been added in DB2 V7 to record DBM1 storage usage statistics. They enable you to monitor the DBM1 address space more effectively so that actions can be taken to alleviate or avoid storage shortage conditions.

IFCID 0225

IFCID 0225 provides you with summary information on the storage usage in the DBM1 address space. It summarizes the detailed information provided by IFCID 0217. This IFCID belongs to Statistics Class 6 and is recorded at the DB2 statistics interval.

You can turn on Statistics Class 6 trace can be turned on with the START TRACE command or the MODIFY TRACE command :

```
-STA TRACE (S) CLASS (1 , 3 , 4 , 5 , 6)  
-MOD TRACE (S) TNO (1) CLASS (1 , 3 , 4 , 5 , 6)
```

IFCID 0217

IFCID 0217 provides you with detailed information on the storage usage in the DBM1 address space. This IFCID is contained in Global Class 10 and is recorded at the DB2 statistics interval.

This record details:

- the amount of available storage in the DBM1 address space
- the amount of storage for MVS use
- the total getmained stack storage
- the total getmained storage

This is followed by information on each DBM1 storage pool, and each agent storage pool. If there are more than 250 such pool entries, they will overflow to another IFCID 0217 record.

For each pool, the total storage used is recorded.

For agent pools, the thread is identified by authorization ID, correlation ID, connection name, and plan name.

IFCIDs 0002, 0003, 0148: page P-lock counters

There are new counters which are used for both Statistics and Accounting traces to record detail on page P-lock requests in a data sharing environment. They record:

- Number of page P-lock requests for space map pages
- Number of page P-lock requests for data pages
- Number of page P-lock requests for index leaf pages
- Number of page P-lock suspensions for space map pages
- Number of page P-lock suspensions for data pages
- Number of page P-lock suspensions for index leaf pages

The Statistics Trace also includes the following additional counters:

- Number of page P-lock negotiations for space map pages
- Number of page P-lock negotiations for data pages
- Number of page P-lock negotiations for index leaf pages

GROUP	TOT4K	CONTINUED	QUANTITY	/SECOND	/THREAD	/COMMIT
-----	-----	-----	-----	-----	-----	-----
PAGE P-LOCK LOCK REQ			115.7K	192.77	N/C	1.49
SPACE MAP PAGES			43724.00	72.88	N/C	0.56
DATA PAGES			0.00	0.00	N/C	0.00
INDEX LEAF PAGES			71929.00	119.89	N/C	0.92
PAGE P-LOCK UNLOCK REQ			114.9K	191.45	N/C	1.47
PAGE P-LOCK LOCK SUSP			2182.00	3.64	N/C	0.03
SPACE MAP PAGES			2017.00	3.36	N/C	0.03
DATA PAGES			0.00	0.00	N/C	0.00
INDEX LEAF PAGES			165.00	0.28	N/C	0.00
PAGE P-LOCK LOCK NEG			1938.00	3.23	N/C	0.02
SPACE MAP PAGES			1898.00	3.16	N/C	0.02
DATA PAGES			0.00	0.00	N/C	0.00
INDEX LEAF PAGES			40.00	0.07	N/C	0.00

IFCIDs 0003, 0147, 0148: global contention

In DB2 V6, the Class 3 wait time for global contention (data sharing only) was reported as a single value.

In DB2 V7 this is now broken down into pairs of elapsed time and event counters:

- Waits for parent L-locks (database, table space, table, or partition)
- Waits for child L-locks (page, or row)
- Waits for other L-locks
- Waits for page set and partition P-locks
- Waits for page P-locks
- Waits for other P-locks

This information is reported in the new Global Contention L-Locks and Global Contention P-Locks blocks of DB2 PM accounting report.

GLOBAL	CONTENTION	L-LOCKS	AVERAGE TIME	AV. EVENT

L-LOCKS			0.002396	0.05
PARENT	(DB, TS, TAB, PART)		0.000000	0.00
CHILD	(PAGE, ROW)		0.001832	0.03
OTHER	0.000000	0.00		

GLOBAL	CONTENTION	P-LOCKS	AVERAGE TIME	AV. EVENT

P-LOCKS			0.000000	0.00
PAGESET/PARTITION			0.000000	0.00
PAGE			0.000000	0.00
OTHER			0.000564	0.03

Real Time Statistics

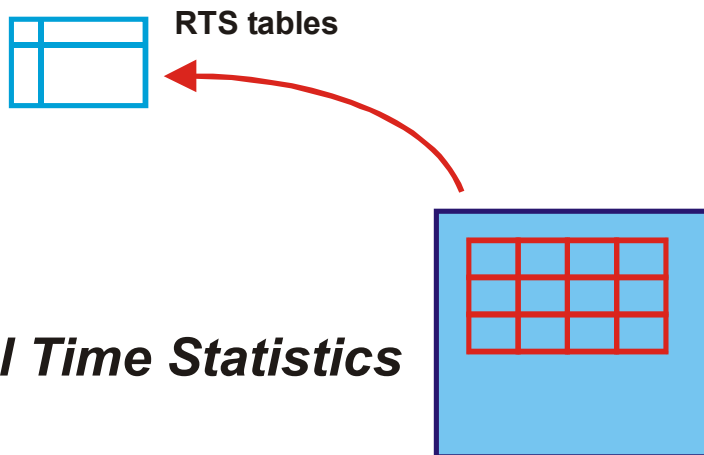
With RTS, DB2 provides the necessary information that end users or automated task schedulers can use to determine which objects require REORG, RUNSTATS or COPY. This will reduce the overall cost of DB2.

Some of the statistics collected are:

- The number of rows, LOB values or index entries modified since last REORG, RUNSTATS or COPY
- The physical space information such as the number of pre-formatted pages, allocated space, and the number of extents
- The number of distinct pages updates and the time of first update since the last COPY

DB2 always generates real time in memory statistics for each table space and index space on your system. Statistics are generated for each partition for partitioned table spaces, and indexes. Optionally, these in-memory statistics can be externalized to DB2 tables from time to time, or when necessary.

You can decide when to run REORG, RUNSTATS and COPY utilities, or when to enlarge your data sets by querying these tables. IBM supplies a sample stored procedure, DSNACCOR, to help you in this task.



Externalizing Real Time Statistics

DPA-230

The Control Center 390 (CC/390) uses the statistics tables. CC/390 queries the DB2 catalog and statistics tables, and identifies the pages sets that require REORG, RUNSTATS or COPY. CC/390 also generates utility statements to perform the utility tasks.

Real time statistics in-memory are **always** generated by DB2, kept in memory, and externalized when necessary. To externalize the statistics, DB2 examines the in-memory statistics, calculates the new totals, updates the new real time statistics tables with the new totals, and resets the in-memory statistics. This process is an asynchronous task. Utilities have an effect upon the statistics, but the changes are synchronous to the utility operations.

When externalizing in-memory statistics, DB2 inserts a row for each partition or non-partitioned page set in SYSIBM.TABLESPACESTATS or SYSIBM.INDEXSPACESTATS. If a row exists it will be updated. The absolute statistic values (for example, TotalRows) are replaced with the new values, and incremental values are summed with the in-memory statistics.

If for some reason DB2 cannot update the statistics tables (perhaps there is lock contention or a resource unavailable) this will not cause the requester to fail. A message (DSNI037I) will be written to the console and a new attempt to externalization will be made in the next update cycle.

The in-memory statistics control block for a page set is usually allocated when it is first created and marked as logically deleted at close. This storage is freed by DB2 after processing the statistics.

SQL INSERT, UPDATE, DELETE, and ROLLBACK statements, and certain utilities cause the in-memory real time statistics to be changed.

You must create the following objects in order to allow DB2 externalizes the in-memory statistics:

Object name Description

DSNRTSDB	Database for real time statistics objects
DSNRTSTS	Table space for real time statistics objects
SYSIBM.TABLESPACESTATS	Table for statistics on table spaces and table space partitions (one row per table space or partition)
SYSIBM.INDEXSPACESTATS	Table for statistics on index spaces and index space partitions (on row per index space or partition)
SYSIBM.TABLESPACESTATS_IX	Unique Index on SYSIBM.TABLESPACESTATS (columns, DBID, PSID, and PARTITION)
SYSIBM.INDEXSPACESTATS_IX	Unique Index on SYSIBM.INDEXSPACESTATS (columns, DBID, PSID, and PARTITION)

DB2 writes the in-memory statistics to real time statistics tables based upon a user-specified time interval which can be modified using a system parameter.

You can set the interval for writing real time statistics on field REAL TIME STATS on panel DSNTIPO of the installation CLIST. It can also be updated by dynamically modifying the system parameter STATSINST. The default value is 30 minutes and the value must be between 1 and 65535.

```

-- Use the following SQL data definition statements to create          00010000
-- the statistics database, table space, tables and indexes.         00020000
--                                                                    00030000
-- The names and declared attributes of the objects must not be changed.00040000
-- However, other attributes can be changed. For example,           00055990
-- COMPRESS YES can be specified.                                    00061980
--                                                                    00070000
-- The amount of primary and secondary space to allocate can be     00080000
-- calculated by using the formulas in the "Administration Guide".   00090000
--                                                                    00100000
-- The approximate number of rows in the two statistics tables       00110000
-- can be determined by the following SQL statement. The sample DDL 00120000
-- used 20,000 objects as an estimate to determine the amount of    00130000
-- space to request.                                                00140000
--                                                                    00150000
-- SELECT C1 + C2 FROM                                             00160000
--     (SELECT COUNT(*) AS C1 FROM SYSIBM.SYSTABLEPART) AS T1,      00170000
--     (SELECT COUNT(*) AS C2 FROM SYSIBM.SYSINDEXPART) AS T2;     00180000
--                                                                    00190000
CREATE DATABASE DSNRTSDB CCSID EBCDIC;                               00200000
CREATE TABLESPACE DSNRTSTS IN DSNRTSDB                             00210000
  CCSID EBCDIC                                                       00220000
  CLOSE NO                                                            00230000
  LOCKMAX 0                                                           00240000
  LOCKSIZE ROW                                                        00244990
  SEGSIZE 32                                                          00250000
  USING STOGROUP SYSDEFLT                                           00260000
  PRIQTY 1600                                                         00270000
  SECQTY 160;                                                         00280000
                                                                    00290000
CREATE TABLE SYSIBM.TABLESPACESTATS                                00300000
  (DBNAME                    CHAR( 8) NOT NULL,                      00310000
  NAME                       CHAR( 8) NOT NULL,                      00320000
  PARTITION                  SMALLINT NOT NULL,                     00330000
  DBID                       SMALLINT NOT NULL,                     00340000
  PSID                       SMALLINT NOT NULL,                     00350000
  UPDATESTATSTIME           TIMESTAMP NOT NULL WITH DEFAULT,       00360000
  TOTALROWS                 FLOAT ,                                  00370000
  NACTIVE                   INTEGER ,                                00380000
  SPACE                      INTEGER ,                                00390000
  EXTENTS                   SMALLINT ,                               00400000
  LOADRLASTTIME             TIMESTAMP ,                              00410000
  REORGLASTTIME             TIMESTAMP ,                              00420000
  REORGINSERTS              INTEGER ,                                00430000
  REORGDELETES              INTEGER ,                                00440000
  REORGUPDATES              INTEGER ,                                00450000
  REORGUNCLUSTINS          INTEGER ,                                00460000
  REORGDISORGLOB            INTEGER ,                                00470000
  REORGMASDELETE           INTEGER ,                                00480000
  REORGNEARINDREF          INTEGER ,                                00490000
  REORGFARINDREF           INTEGER ,                                00500000
  STATSLASTTIME             TIMESTAMP ,                              00510000
  STATSINSERTS              INTEGER ,                                00520000
  STATSDELETES              INTEGER ,                                00530000
  STATSUPDATES              INTEGER ,                                00540000
  STATSMASDELETE           INTEGER ,                                00550000
  COPYLASTTIME              TIMESTAMP ,                              00560000
  COPYUPDATEDPAGES          INTEGER ,                                00570000
  COPYCHANGES              INTEGER ,                                00580000
  COPYUPDATELRSN            CHAR(6) FOR BIT DATA,                 00590000
  COPYUPDATETIME            TIMESTAMP )                              00600000
  IN DSNRTSDB.DSNRTSTS CCSID EBCDIC;                                00610000
                                                                    00620000

```

```

CREATE UNIQUE INDEX SYSIBM.TABLESPACESTATS_IX          00630000
ON SYSIBM.TABLESPACESTATS                            00640000
  (DBID, PSID, PARTITION)                            00650000
CLUSTER CLOSE NO;                                   00660000
                                                    00670000
CREATE TABLE SYSIBM.INDEXSPACESTATS                 00680000
(DBNAME          CHAR( 8) NOT NULL,                  00690000
 INDEXSPACE      CHAR( 8) NOT NULL,                  00700000
 PARTITION       SMALLINT NOT NULL,                  00710000
 DBID            SMALLINT NOT NULL,                  00720000
 ISOBID          SMALLINT NOT NULL,                  00730000
 PSID           SMALLINT NOT NULL,                  00740000
 UPDATESTATTIME  TIMESTAMP NOT NULL WITH DEFAULT,   00750000
 TOTALENTRIES    FLOAT                               ,   00760000
 NLEVELS         SMALLINT                            ,   00770000
 NACTIVE         INTEGER                             ,   00780000
 SPACE           INTEGER                             ,   00790000
 EXTENTS         SMALLINT                            ,   00800000
 LOADRLASTTIME   TIMESTAMP                           ,   00810000
 REBUILDLASTTIME  TIMESTAMP                          ,   00820000
 REORGLASTTIME    TIMESTAMP                          ,   00830000
 REORGINSERTS     INTEGER                             ,   00840000
 REORGDELETES     INTEGER                             ,   00850000
 REORGAPPENDINSERT  INTEGER                           ,   00860000
 REORGPSEUDODELETES  INTEGER                          ,   00870000
 REORGMASSDELETE   INTEGER                            ,   00880000
 REORGLAFAFFAR     INTEGER                            ,   00890000
 REORGLAFAFFAR     INTEGER                            ,   00900000
 REORGLAFAFFAR     INTEGER                            ,   00910000
 REORGLAFAFFAR     INTEGER                            ,   00920000
 REORGLAFAFFAR     INTEGER                            ,   00930000
 REORGLAFAFFAR     INTEGER                            ,   00940000
 REORGLAFAFFAR     INTEGER                            ,   00950000
 REORGLAFAFFAR     INTEGER                            ,   00960000
 REORGLAFAFFAR     INTEGER                            ,   00970000
 REORGLAFAFFAR     INTEGER                            ,   00980000
 REORGLAFAFFAR     CHAR(6) FOR BIT DATA,           00990000
 REORGLAFAFFAR     INTEGER                            ,   01000000
 REORGLAFAFFAR     INTEGER                            ,   01010000
 IN DSNRTSDB.DSNRTSTS CCSID EBCDIC;                 01020000
                                                    01030000
CREATE UNIQUE INDEX SYSIBM.INDEXSPACESTATS_IX       01040000
ON SYSIBM.INDEXSPACESTATS                           01050000
  (DBID, ISOBID, PARTITION)                          01060000
CLUSTER CLOSE NO;

```

Starting the RTS database

START DATABASE(DSNRTSDB)

After you create the RTS database, DB2 immediately puts it into a stopped state. When the database is explicitly started, DB2 enables the externalization of the real time statistics. During this start database process, the statistics objects are validated and if they are correct, DB2 enables the process to externalize in-memory statistics.

At the end of STATSINT interval all in-memory statistics are written to tables.

-STOP DATABASE(DSNRTSDB)

-STOP DATABASE(db-name) SPACENAM(space-name)

The externalize process is controlled by the RTS manager. It runs under a system task in DBM1 address space that is created during DB2 start up. The RTS manager is triggered on a time interval defined by the system parameter STATSINT. It carries out the following tasks:

- Places the active statistics blocks in clustering order
- Inserts/updates the rows in the RTS tables through the clustering index
- Frees any dormant statistics blocks that belong to data sets being closed

INSERT, UPDATE, and DELETE statements cause DB2 to modify the real time statistics. In addition, certain DB2 utilities also affect the statistics.

The real time statistics are normally accurate values. However, several factors can cause the tables values to become inaccurate:

- Some utility restart scenarios
- DB2 subsystem failure
- DB2 stopped with STOP DB2 MODE(FORCE)
- Notify failure in a data sharing environment
- Running third party utilities without flushing the in-memory statistics

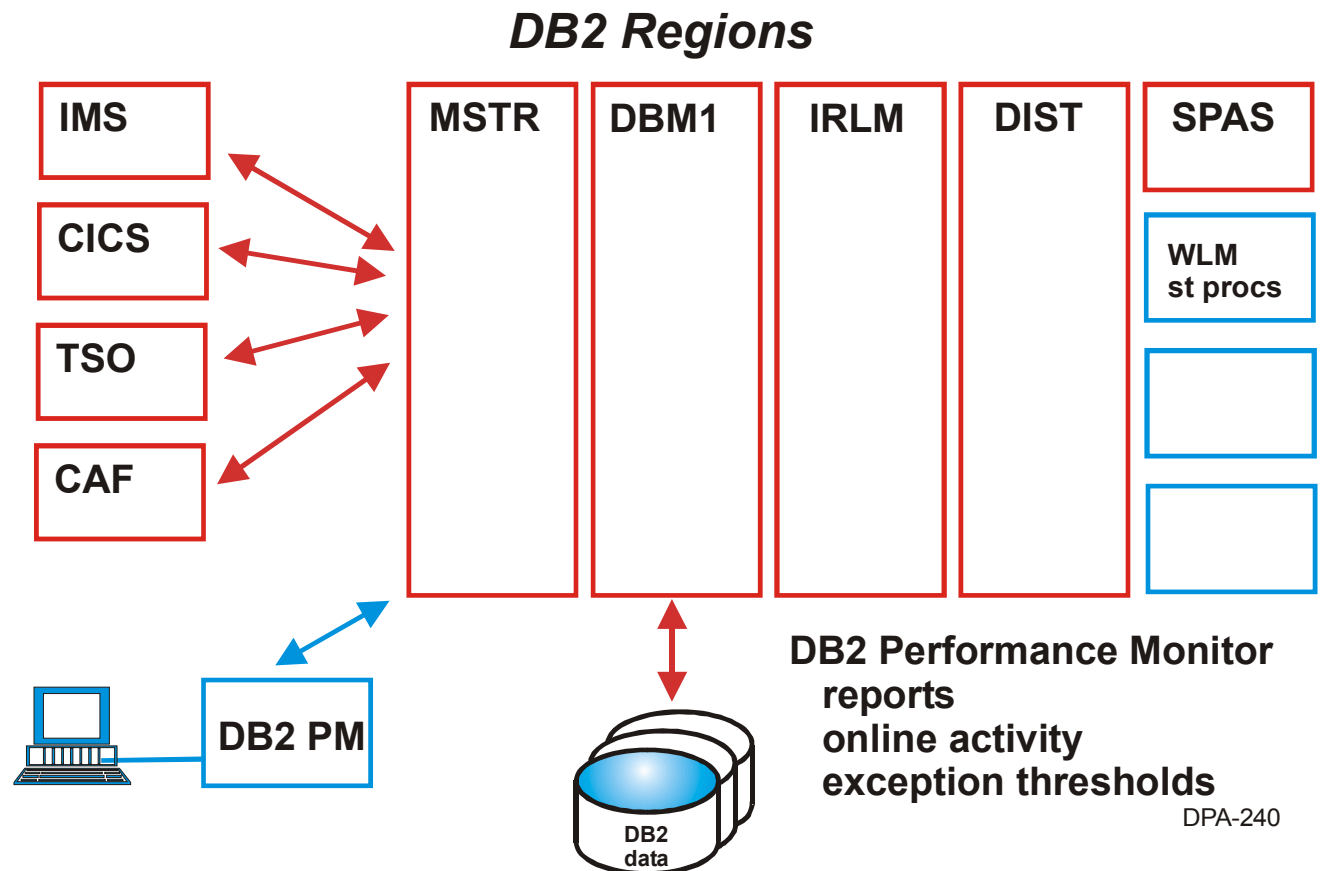
To restore accurate values, you will need to run REORG, RUNSTATS and COPY on the suspect object.

Hints and tips

- put the RTS objects in an isolated buffer pool. This will increase the performance of the RTS manager accessing and updating these objects.
- use uncommitted read (UR) lock isolation whenever possible to avoid time-outs and deadlocks when accessing RTS tables with RTS manager. Avoid using SHRLEVEL CHANGE when running REORG, RUNSTATS and COPY on the RTS objects.
- The impact on CPU of the code increase in DB2 due to incrementing and decrementing row, LOB, or index counts has been measured as less than 5%. IBM calls this minimal.

DB2 Performance Monitor

DB2 PM is the principal tool for system and application monitoring. Other tools are additional, complementary products providing low cost and easy application information. For detailed application tracing, DB2 PM should be used.



DB2 PM batch reporting is the best IBM tool for DB2 performance. It is very accurate, very flexible and provides a wide variety of reports. Its LAYOUT LONG versions of accounting and statistics reports DB2 PM provide much information for detailed servicing and debugging.

IBM DATABASE 2 Performance Monitor for OS/390

Command ==> _____

Select one of the following.

- ___ 1. Create and execute DB2 PM commands
- 2. Display and print graphs
- 3. View online DB2 activity
- 4. Maintain parameter data sets
- 5. Customize DB2 PM report and trace layouts
- 6. Exception profiling

IBM DB2 UDB Performance Monitor for OS/390 V6
Licensed Materials - Property of IBM
5645-DB2 (C) Copyright IBM Corp. 1985, 1998.
All rights reserved.
US Government Users Restricted Rights -
Use, duplication or disclosure restricted
by GSA ADP Schedule Contract with IBM Corp.

03/01/13 16:56 DB2 PM Online Monitor Main Menu UKEXTHD2TA D2TA V6
Command ==> _____

Select one of the following.

- ___ 1. Display Thread Activity
- 2. Display Statistics
- 3. Display System Parameters
- 4. Options
- 5. Control Exception Processing
- 6. Collect Report Data
- 7. IRF - Create and execute DB2 PM commands
- 8. IRF - Display and print graphs
- 9. IRF - Maintain parameter data sets
- 10. Explain

03/01/13 17:23

DB2 Statistics Detail

UKEXTHD2TA D2TA V6

Command ==>

For details, type any character next to heading, then press Enter.

	More:	+
EDM Pool		
- EDM Pool full	:	0
EDM Pool pages in use (%)	:	40.7
CT requests/CT not in EDM pool	:	53.0
PT requests/PT not in EDM pool	:	6.5
DBD requests/DBD not in EDM pool	:	35.8
Buffer Manager		
- Synchronous Reads	:	649440
Deferred write threshold reached	:	0
DM critical threshold reached	:	
Locking Activity		
- Suspensions - all	:	4821
Deadlocks	:	0
Timeouts	:	0
Lock escalations - all	:	0
Open/Close Management		
- Open data sets - High Water Mark	:	482
Bind Processing		
Plan/Package Allocation, Authorization Management		
Log Manager		
- Reads satisfied - Output Buffer	:	9225
Reads satisfied - Active Log	:	1168
Reads satisfied - Archive Log	:	0
Write-no-wait	:	119554
Unavailable output log buffers	:	0
Subsystem Service		
- Queued at create thread	:	0
System event checkpoints	:	4
SQL Activity		
PREPARE Detail		
- Prepare	:	1091
Query Parallelism Data		
RID List Processing		
Distributed Data		
CPU Times and Other Data		
Data Sharing Locking Activity		
Group Buffer Pool Activity		
Global Group Buffer Pool Statistics		
Stored Procedures		

Query Monitor

IBM DB2 Query Monitor is a low overhead thread monitoring tool which can concurrently monitor up to 64 DB2 subsystems with information filters and exception threshold processing. All dynamic and static SQL that is issued is reported. DB2 commands that have been issued are also detected and reported.

It needs the DB2 accounting trace to be started, with classes 1, 2 and 3 selected. The historical data is stored in DB2 tables.

Information on SQL statements includes:

- CPU time
- elapsed time
- buffer pool statistics
- I/O delays
- locking

Query Monitor complements the functions of DB2 Performance Monitor (DB2 PM); Query Monitor (QM) provides information quickly, DB2 PM can then be used to gather more detailed information and perform a more in-depth investigation.

You can set thresholds to highlight threads which have exceeded certain customizable limits so that quick problem detection is possible via the online panels.

The main features of QM:

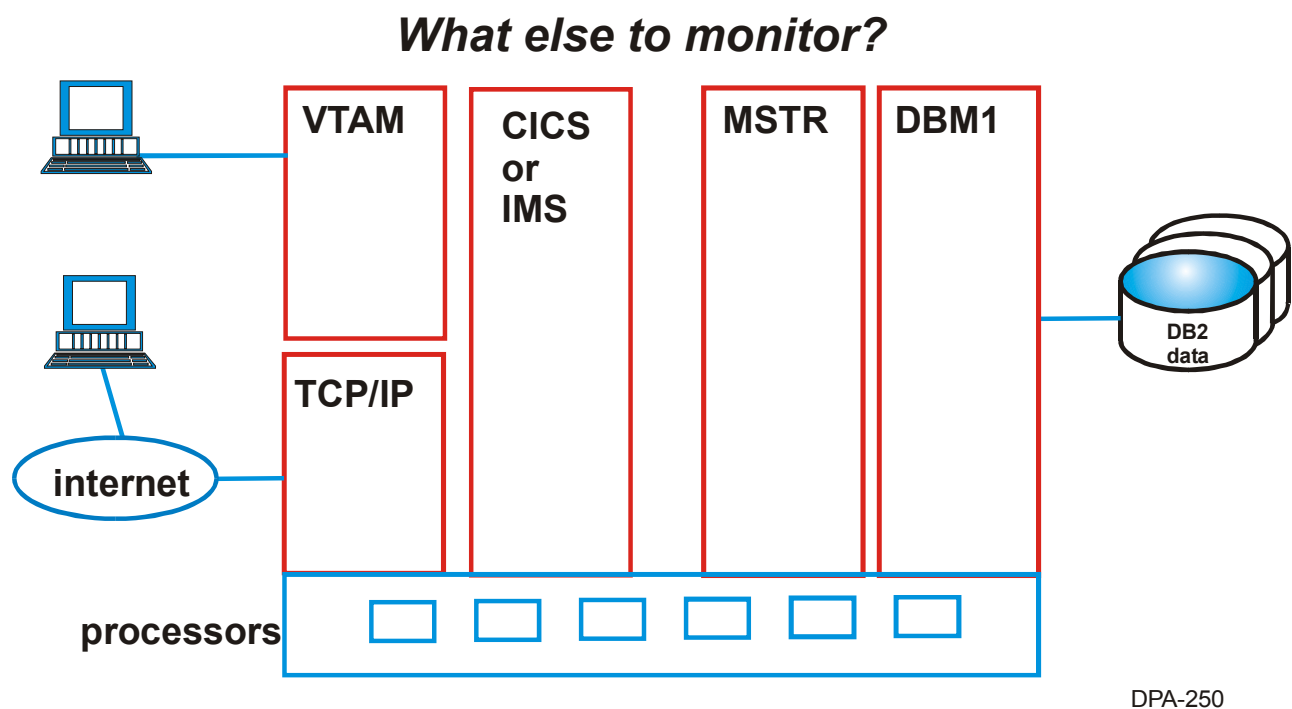
- Ease of use
ISPF interface with extensive online help.
Color coding of active and exception threads.
- Ability to monitor up to 64 DB2 subsystems
- Low system overhead
- Exception processing
Threads which have exceeded pre-determined limits are flagged in red.
- Dynamic filtering:
“Information overload” can be eliminated by using application profiles to dynamically reduce the information displayed by use of selective filtering.
- Historical data
Retain historical data on disk or tape or for subsequent loading to DB2 tables for analysis.

What else to monitor?

Network response

The response from the processor is likely to be measured in fractions of a second, but responses in networks are measured in seconds. A poorly performing or overloaded network will always be a significant aspect of response time no matter how fast the processor.

Distributed system transactions slow response further and might need some data distribution design to reduce the crossing of platforms.



Disk response

I/O operations are responsible for the major portion of internal processing time for a transaction. Consider actions to reduce the I/O access per transaction to a minimum. Increasing the usage of DB2 buffer pools and data spaces backed up by memory will reduce I/O.

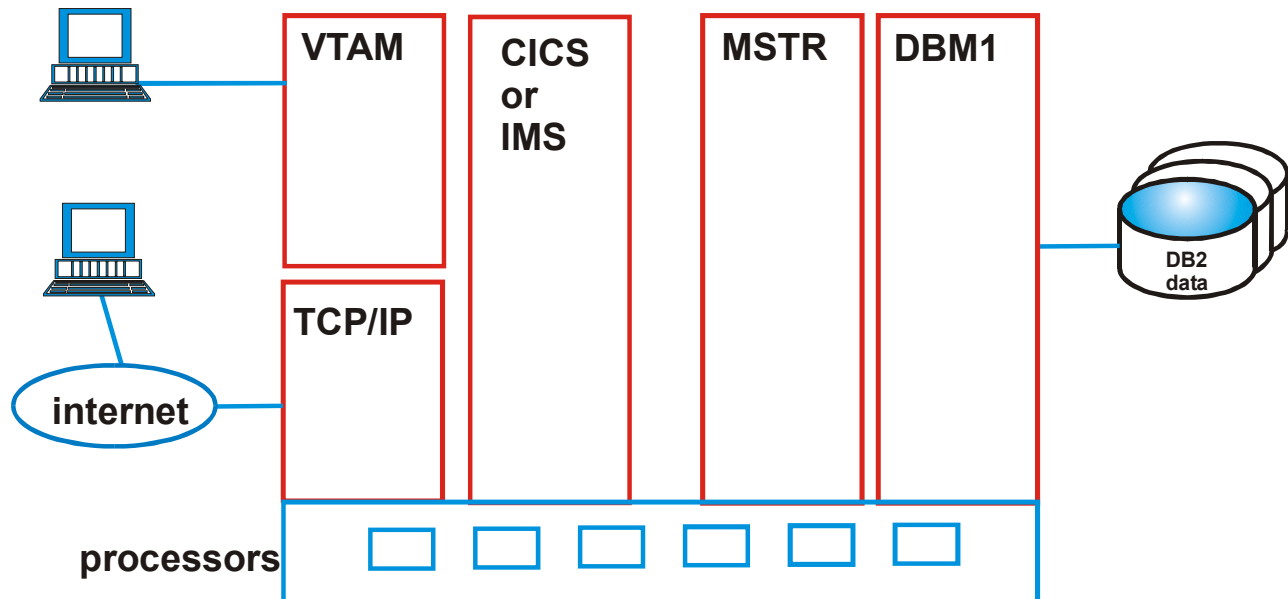
Response times

Response times are difficult to predict before design has been completed. Therefore plan to review response time frequently. Be aware that distributed processing adds an overhead at both the local and remote sites.

Workload peaks

Aim to base the estimates on peak workloads, paying special attention to their distribution. Do not forget periodic batch applications such as month-end processing. You should also consider downtime for maintenance as peaks will exist following such activity.

What else to monitor?



DPA-250

Service levels

Service Level Agreements often include:

- Expectations of query response time
- Transaction throughput
- Batch throughput
- Housekeeping schedules

Workload types

It is important for you to recognize the different types of workload that exist within a system. You should examine the workload and group transactions by their function. Some transactions perform the same type of function and have a common identifiable workload profile. Other transactions, for example, QMF queries or accesses from the Web may be diverse by their nature and difficult to group.

Planning for performance

Initially, plan to gather the resource requirements by estimating the following:

- Transactions
- Maximum rate of transaction per minute, per hour, per day
- Number of I/O operations per transaction
- Average and maximum processor usage per transaction
- Size of tables
- Table space definition
- Index key distribution
- Support personnel

Two products (the SQL/Performance Analyzer and DB2 Estimator) can help in estimating the overall processor usage per transaction.

- Query times
- Online query processing load
- Canned or user-written queries

SQL/PA and DB2 Estimator can help in estimating the CPU and I/O resources required for queries.

- Batch processing
- Length of batch window
- Batch processing load
- Size of tables
- Housekeeping routines

SQL/PA

SQL/PA is a simple tool that addresses the needs of fast development and software control. It does not require DB2 tracing activity, nor the actual execution of the SQL statement, and therefore does not add any extra workload to the DB2 system.

The results SQL/PA produces are *estimates* and could be different from the real execution measurements, in fact SQL/PA's strength is to provide an indication of poor performance and to advise about correction, not to report actual execution measurements.

SQL/PA does not compete with DB2 PM, but it offers the opportunity to develop better applications while assisting less experienced staff to walk through the intricacies of DB2 performance.

DB2 Estimator

This is similar to SQL/PA. DB2 Estimator provides a performance modeling capability but it is not as easy to use. The model definition in DB2 Estimator is almost entirely manual and therefore can be time-consuming to set up. It requires download of the catalog information and the SQL.

Conversely SQL/PA runs directly on the host where DB2 and your applications and catalog are, and there is very minimum manual intervention to define the models. In addition SQL/PA provides governing capability, warnings and guidelines through SQL Advisor and Explain information in a sentence-like style.