

# 1 All about DB2 z/OS Traces

DB2<sup>1</sup> has extensive tracing facilities that generate trace records called IFCID's. IFCID's are produced by the DB2 Instrumentation Facility. These trace records describe internal events within DB2 at various levels of granularity. The Instrumentation Facility Interface allows application programs to retrieve IFCID's from DB2. IFCID's have a complex record format, and require sophisticated programming techniques for interpretation. The DB2 START TRACE command controls the production and disposition of IFCID's. IFCID's are useful for analysis and planning purposes. IFCID's can be written to SMF, GTF, or OP buffer destinations. They can be processed by vendor report packages, or by locally developed SAS programs. Depending on the type of trace and the classes of information collected, DB2 traces can introduce overhead that ranges from inconsequential to devastating.

## Table of Contents:

1.1	Introduction .....	2
1.2	Types of trace, trace classes and IFCID's .....	2
1.2.1	Statistic Traces .....	2
1.2.2	Accounting Traces .....	4
1.2.3	Audit Traces .....	6
1.2.4	Performance Traces .....	8
1.2.5	Monitor Traces .....	9
1.2.6	Global Traces .....	10
1.3	Trace Overhead .....	11
1.3.1	Minimize Trace Overhead .....	11
1.4	Controlling Traces .....	13
1.4.1	Starting a trace .....	14
1.4.2	Stopping a trace .....	15
1.4.3	Modifying a trace .....	15
1.4.4	Displaying a trace .....	15
1.4.5	DSN1SDMP – selective dumps .....	15
1.4.6	DB2 Utilities with DIAGNOSE .....	17
1.4.7	Authorization .....	18
1.5	Trace Record destinations .....	18
1.6	RMID's .....	20
1.7	How it works internally .....	20
1.8	DSN1SMFP - printing SMF records .....	21
1.9	DSNTSMFD - SMF Decompression .....	22
1.10	All traces, classes and IFCID's .....	23
1.11	Loading IFCID description into a table .....	27
1.11.1	Create Tablespace and Tables .....	28
1.11.2	Loading IFCID descriptions and Trace types .....	29
1.12	Tools processing DB2 Trace data .....	29
1.12.1	Format of Trace Records .....	29
1.12.2	BMC Mainview for DB2 z/OS and DB2 Traces .....	30
1.12.3	BMC Performance Reporter .....	31
1.12.4	Compuware Strobe for DB2 .....	33
1.12.5	OMEGAMON XE for DB2 PE .....	34
1.12.6	SAP and DB2 for z/OS .....	35

<sup>1</sup> This documentation covers DB2 for z/OS Version 9, DB2 10 for z/OS and DB2 11 for z/OS. DB2 ® and z/OS ® are property and registered trademarks of the IBM ® Corporation. All information can be found in the appropriate program product documentation.

1.13	List of available IFCID's .....	36
1.14	IFCID mappings .....	38
1.15	Changes from DB2 for z/OS Version 10.....	39
1.15.1	SMF subtype field defined as a two-byte value .....	39
1.15.2	Changed default OP buffer size.....	39
1.15.3	New IFCID's.....	39
1.15.4	Changed IFCID's .....	40
1.15.5	Changed format of the IFCID 0002, 0003, 0148, and 0306 trace records .....	43
1.15.6	Changes in IFCID structure documented with OMEGAMON .....	43
1.16	Changes from DB2 11 for z/OS.....	43
1.16.1	New IFCID's.....	43
1.16.2	Changed IFCID's .....	43
1.17	DB2 Monitoring Glossary.....	45

## 1.1 Introduction

Significant internal events within DB2 are represented by trace data created by DB2 resource managers and collected by the DB2 Instrumentation Facility. This facility is itself a fully integrated DB2 resource manager responsible for the collection and distribution of trace data. The Instrumentation Facility organizes event data into logical records called IFCID's. These IFCID's are recorded as variable-length, self-defining physical records that can be processed by QSAM or BSAM.

The DB2 Instrumentation Facility (IFI) is a software interface through which external programs may request and obtain IFCID's from DB2. Application monitor programs may connect to DB2 via the batch Call Attach Facility. Once such a connection (thread) is established, DB2 traces can be started and stopped under program control.

While a DB2 trace is active, an application can collect IFCID's from DB2 by issuing READA and READS IFI calls. Each kind of call acquires its own kind of data. READS calls collect snapshots of work in progress, and statistical data about the system. READA calls collect comprehensive low level detail IFCID's.

## 1.2 Types of trace, trace classes and IFCID's

DB2 trace can record six types of data:

- **statistics:** Statistical data to evaluate the performance of the system.
- **accounting:** Who uses the system.
- **audit:** Who performs changes or access data, who changes authorization, who fails to do.
- **performance:** Provides deeper insight view to analyze particular situations.
- **monitor:** Provides data to your monitor software product.
- **and global:** Mainly addressed to IBM service personal.

### 1.2.1 Statistic Traces

The statistics trace reports information about how much the DB2 system services and database services are used. Statistics trace is a system-wide trace and must not be used for charge-back accounting. Use the information the statistics trace provides to plan DB2 capacity, or to tune the entire set of active DB2 programs.

- Class 1 provides information about system services, database statistics, and statistics for the DBM1 address space. It also includes the system parameters that were in effect when the trace was started.
- Class 3 provides information about deadlocks and timeouts.

- Class 4 provides information about exceptional conditions.
- Class 5 provides information about data sharing.
- Class 6 provides storage statistics for the DBM1 address space.

The statistics trace is written on an interval basis, and you can control the exact time that statistics traces are taken.

STATISTICS trace output (IFCID's 001 and 002) is useful for trend analysis and capacity planning considerations. These IFCID's can be directed to SMF. During DB2 installation, a DSNZPARM specification is made on panel DSNTIPN STATISTICS TIME field (STATIME) that determines how frequently the system statistics trace records are produced. Since STATISTICS IFCID's are only produced periodically, they are not a continuous source of internal overhead.

DB2 10 always generates an SMF type 101 trace record (statistics trace) every one minute interval, no matter what you specify in the STATIME DSNZPARM parameter. This trace interval should not severely impact SMF data volumes, because only 1359 records are produced per DB2 subsystem per day. The STATIME subsystem parameter applies only to IFCIDs 0105, 0106, 0199, and 0365.

In DB2 Version 11, the STATIME subsystem parameter applies only to IFCIDs 0105, 0106, 0199, and 0365. IFCIDs 0001, 0002, 0202, 0217, 0225, and 0230 are no longer controlled by STATIME, and the corresponding trace records are written at fixed, one-minute intervals.

DB2 10 restructures IFCID 225 to take into account the DB2 10 memory mapping and 64-bit addressing. The trace record is also now divided into data sections for a more logical grouping of data. IFCID 225 also contains information about storage in the DIST address space. IFCID 217 is also outdated. Duplicate data with IFCID 225 is now removed. Enable both IFCID 225 and IFCID 217 to generate a detail system storage profile. However, in most cases IFCID 225 is sufficient for general monitoring and reporting.

In DB2 Version 11, some fields in IFCID 0002 and IFCID 0225 that are related to shareable storage have been expanded from 4 to 8 bytes, moved, and renamed.

For more details on changes which came with DB2 10 see "Changes from DB2 for z/OS Version 10" on page 39. For more details on changes which came with DB2 11 see "Changes from DB2 for z/OS Version 10" on page 43.

Type	Class	Data collected	IFCID's activated
Statistics	1	statistical data	1-2,105,106, 202
	2	installation-defined statistics record	152
	3	deadlock and lock timeout information, connect or disconnect from a group buffer pool, long-running UR's	172,196,250, 261,262,313, 258, 335, 337
	4	DB2 exception conditions	173,191,192, 193,194,195, 203,204,205, 206,207,208, 209,210,235, 236,(238), 267,268,343
	5	data sharing global statistics	230,254
	6	DBM1 storage summary	0225
	7	DB2 V10 statistics about communication with DRDA remote locations	365 (V10)
	8	buffer pool data set statistics	0199
	9	aggregate CPU and wait time statistics by connection type	0369 (V11)
	10 -29	reserved	
	30 -32	available for local use	

For a complete list of available IFCID's see section "List of available IFCID's" on page 36.

## 1.2.2 Accounting Traces

The accounting trace is used to associate resource consumption with a consumer (thread) using application programs. Many levels of trace granularity are available according to the accounting classes specified on the START TRACE command. An accounting trace provides the following types of information:

- Start and stop times
- Number of commits and aborts
- The number of times certain SQL statements are issued
- Number of buffer pool requests
- Counts of certain locking events
- Processor resources consumed
- Thread wait times for various events
- RID pool processing
- Distributed processing
- Resource limit facility statistics

DB2 trace begins collecting these items when a thread is successfully allocated to DB2. DB2 trace writes a completed record when the thread terminates or when the authorization ID changes.

The Thread Accounting record (IFCID 003) is built at thread completion, and is a summary of resources used by the thread. Accounting trace IFCID's can be directed to SMF for processing by post processor accounting systems. An accounting trace produces few records, and introduces a small internal overhead to DB2.

The terminology of "DB2 Accounting Time":

<b>plan</b>	<p><b>Class 1 times</b></p> <ul style="list-style-type: none"> <li>• Application time from connect to DB2 (thread creation) till disconnect (thread termination)</li> <li>• Both class 1 elapsed time and class 1 CPU time are reported.</li> </ul> <p><b>Class 2 times</b></p> <ul style="list-style-type: none"> <li>• Time spent within DB2</li> <li>• Both class 2 elapsed time and class 2 CPU time are reported.</li> </ul> <p><b>Class 3 times</b></p> <ul style="list-style-type: none"> <li>• Thread suspension time, e.g. for synchronous I/O</li> </ul>
	<p><b>Class 5 times</b></p> <ul style="list-style-type: none"> <li>• Time spent for IFI calls.</li> <li>• Both class 5 elapsed time and class 5 CPU time are reported.</li> </ul>
<b>package</b>	<p><b>Class 7 times</b></p> <ul style="list-style-type: none"> <li>• Like class 2, but on package/DBRM level.</li> <li>• Both class 7 elapsed time and class 7 CPU time are reported.</li> </ul> <p><b>Class 8 times</b></p> <ul style="list-style-type: none"> <li>• Like class 3, but on package/DBRM level.</li> </ul>

(See also "DB2 Monitoring Glossary" on page 45.)

### CICS and thread reuse:

When a thread is reused without initiating sign-on, several transactions are accumulated into the same accounting record. Applications that use ACCOUNTREC(UOW) or ACCOUNTREC(TASK) in the DBENTRY RDO definition initiate a partial sign-on, which creates an accounting record for each transaction.

### Package Information:

Before you can turn on accounting for packages and DBRMs, accounting trace class 7 or 8 must be active. Activate accounting trace class 8 to collect information about the amount of time an agent was suspended in DB2 for each executed package. If accounting trace classes 2 and 3 are activated, activating accounting trace classes 7 and 8 incurs minimal additional performance cost.

### ACCUMAC DB2 Installation parameter:

Accounting trace records are affected by the DB2 subsystem parameter ACCUMACC, which controls whether and when DB2 Accounting data is accumulated by the user for DDF and RRSAF threads. A parameter value of 2 or greater causes Accounting records to roll up into a single record every n occurrences of the user on the thread. These values can be set by DDF threads by Server Connect and Set Client calls, and by RRSAF threads by RRSAF SIGN, AUTH SIGNON, and CONTEXT SIGNON functions. When roll-up occurs, the values of some fields shown in Accounting reports and traces lose their meanings because of the accumulation. Thus, these fields are often marked as N/A, N/P or N/C for derived fields.

### PTASKROL DB2 Installation parameter:

By default, Accounting trace records from parallel query tasks are rolled up into the originating task's Accounting trace. This is controlled by the DB2 subsystem parameter PTASKROL (parallel tasks roll-up), which has a default value of YES. A roll-up record is written when the parent task (agent) deallocates on an originating DB2, or when an accumulating child task is deallocated on an assisting DB2. The rolled up data is an accumulation of all counters for that field for each child task that completed and deallocated. When roll-up occurs, the values of some fields in Accounting reports and traces lose their meanings because of the accumulation. Thus, these fields are often marked as N/A, N/P, or N/C for derived fields.

Type	Class	Data collected	IFCID's activated
Accounting	1	accounting data	3,106
	2	in DB2 time	200, 232
	3	wait time for i/o, locks, latches, drains, and claims, asynchronous GBP requests	6-9,32-33, 44-45,(51-52, 56-57),117-118, 127-128,170, 171, 174-175, 213-214, 215-216, 226-227, 242-243, 329, 351, 352, 382-383
	4	installation-defined accounting record	151
	5	time spent processing IFI requests	187
	7	package level accounting in-DB2 time	232,239,240
	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, 239, 241-243, 351, 352, 382-383
	10	package detail	239
	11	Plan-level only	Nnn (V11)
	12 -29	reserved	
	30 -32	available for local use	

For a complete list of available IFCID's see section "List of available IFCID's" on page 36.

Requesting accounting trace data is verify special in that, starting additional trace classes don't create extra trace records, they just add counters to existing records. The result of that is that if ...

```
-STA TRA(A) C(1) DEST(SMF)
-STA TRA(A) C(1,2,3) DEST(OPX)
```

the SMF records will also contain the class 2 and 3 data. The record is only built once in storage.

### 1.2.3 Audit Traces

The audit trace collects information about DB2 security controls and is used to ensure that data access is allowed only for authorized purposes. The default causes no auditing to take place.

On the CREATE TABLE or ALTER TABLE statements, you can specify whether or not a table is to be audited, and in what manner; you can also audit security information such as any access denials, grants, or revokes for the table.

If you specified YES for AUDIT TRACE (AUDITST) on installation panel DSNTIPN, audit trace class 1 starts automatically when you start DB2.

The audit trace is used to supervise the access to objects by users. Audit IFCID's are often sent to SMF. IFCID's produced by the AUDIT trace contain detailed information about the use of DB2 objects, and are used by security analysis software packages for system administration purposes. An audit traces monitor specific thread related events that may not repeat very often during the life of a given thread. However, since the AUDIT trace tracks all threads in the system, there may be a moderate overhead introduced by the audit trace. (For more details on possible overhead, see "1.3 Trace Overhead" on page 11.)

The audit trace has certain limitations, including that it does not automatically record everything. The audit trace has the following additional limitations:

- The audit trace must be turned on; it is not on by default, but DB2 may start audit trace on start-up.
- The trace does not record old data after it is changed.
- If an agent or transaction accesses a table more than once in a single unit of recovery, the audit trace records only the first access.
- The audit trace does not record accesses if you do not start the audit trace for the appropriate class of events.
- Except class 8, the audit trace does not audit certain utilities. For example, the trace audits the first access of a table with the LOAD utility, but it does not audit access by the COPY, RECOVER, and REPAIR utilities. The audit trace does not audit access by stand-alone utilities, such as DSN1CHKR and DSN1PRNT.
- The trace audits only the tables that you specifically choose to audit.
- You cannot audit access to auxiliary tables.
- You cannot audit the catalog tables because you cannot create or alter catalog tables.

This auditing coverage is consistent with the goal of providing a moderate volume of audit data with a low impact on performance. However, when you choose classes of events to audit, consider that you might ask for more data than you are willing to process.

Type	Class	Data collected	IFCID's activated
Audit	1	authorization failures Access attempts that DB2 denies because of inadequate authorization. This class is the default.	140
	2	explicit grant and revoke Explicit GRANT and REVOKE statements and their results. This class does not trace implicit	141

Type	Class	Data collected	IFCID's activated
		grants and revokes.	
	3	<p>create, drop, and alter operations against audit tables</p> <p>Traces CREATE, DROP, and ALTER operations against an audited table or a table that is enabled with multilevel security with row-level granularity. For example, it traces the updates to a table created with the AUDIT CHANGES or AUDIT ALL clause. It also traces the deletion of a table as the result of a DROP TABLESPACE or DROP DATABASE statement.</p>	142
	4	<p>first change of audited object</p> <p>Changes to audited tables. Only the first attempt to change a table, within a unit of recovery, is recorded. (If the agent or the transaction issues more than one COMMIT statement, the number of audit records increases accordingly.) The changed data is not recorded; only the attempt to make a change is recorded. If the change is not successful and is rolled back, the audit record remains; it is not deleted. This class includes access by the LOAD utility. Accesses to a dependent table that are caused by attempted deletions from a parent table are also audited. The audit record is written even if the delete rule is RESTRICT, which prevents the deletion from the parent table. The audit record is also written when the rule is CASCADE or SET NULL, which can result in deletions that cascade to the dependent table.</p>	143
	5	<p>first read of audited object</p> <p>All read accesses to tables that are identified with the AUDIT ALL clause. As in class 4, only the first access within a DB2 unit of recovery is recorded. References to a parent table are also audited.</p>	144
	6	<p>SQL statement at bind</p> <p>The bind of static and dynamic SQL statements of the following types:</p> <ul style="list-style-type: none"> <li>• INSERT, UPDATE, DELETE, CREATE VIEW, and LOCK TABLE statements for audited tables. Except for the values of host variables, the audit record contains the entire SQL statement.</li> <li>• SELECT statements on tables that are identified with the AUDIT ALL clause. Except for the values of host variables, the audit record contains the entire SQL statement.</li> </ul>	145
	7	<p>change in authorization</p> <p>Assignment or change of an authorization ID because of the following reasons:</p> <ul style="list-style-type: none"> <li>• Changes through an exit routine (default or user-written)</li> <li>• Changes through a SET CURRENT</li> </ul>	55, 83,87,169, 319

Type	Class	Data collected	IFCID's activated
		SQLID statement <ul style="list-style-type: none"> <li>An outbound or inbound authorization ID translation</li> <li>An ID that is being mapped to a RACF ID from a Kerberos security ticket</li> </ul>	
	8	utility access to any object The start of a utility job, and the end of each phase of the utility.	23,24,25,219, 220
	9	installation-defined audit record Various types of records that are written to IFCID 0146 by the IFI WRITE function.	146
	10	trusted context audit CREATE and ALTER TRUSTED CONTEXT statements, establish trusted connection information and switch user information.	269,270
	11	DB2 V10 extended authorization checking	270, 361, 362,
	12 -29	reserved	
	30 -32	available for local use	

For a complete list of available IFCID's see section "List of available IFCID's" on page 36.

## 1.2.4 Performance Traces

The performance trace provides information about a variety of DB2 events, including events related to distributed data processing. You can use this information to further identify a suspected problem, or to tune DB2 programs and resources for individual users or for DB2 as a whole.

Performance trace can capture more kinds of IFCID's than any other type of DB2 trace. The collection granularity varies from coarse to extremely fine. Because the potential output from the Performance trace is so great, IBM recommends that the IFCID's be directed to GTF.

The performance trace can be tailored to monitor very specific events, or can be unrestricted for a system-wide view. This practice requires careful specification of the information to be collected.

A utility thread is identified by the presence of IFCID's 023, 024, and 025. The DB2 utility "name" (COPY, REORG, etc) is externalized via IFCID 23.

IFCID 25 in performance class 10 trace is enhanced to provide information on the CPU time used by the utility at termination as well as the CPU time spent in sort processing by that utility. When accounting class 1 trace is active in DB2 then the time spent on zIIP by the utility will be written to IFCID 25. (See APAR PM37956: UTILITY ACCOUNTING ENHANCEMENT).

Type	Class	Data collected	IFCID's activated
Performance	1	background event	1-2,31,42-43, 76-77,78-79, 102,103,105, 106,107,153
	2	subsystem related events	3,68-69, 70-71,72-73, 74-75,80-81, 82-83,84-85, 86-87,88-89, 106,174-175, 321,322
	3	SQL-related events	173,22,53,55, 58,59,60,61, 62,63,64,65, 66,92,95-96, 97,106,112, 177,233,237, 272,311,324, 343, 360
	4	buffer manager i/o and EDM pool requests	6-7,8-9-10, 29-30,105, 106,107,127- 128, 226-227, 357, 358, 359
	5	log manager	32-33,34-35, 36-37,38-39, 40-41,104,106 114-115-116, 117-118,119- 120,(228-229)

Type	Class	Data collected	IFCID's activated
	6	summary lock information	20,44-45, 105,106,107, 172,196,213-214,218, 337
	7	detailed lock information	21,105,106, 107,223
	8	data manager detail	13-14,15-16- 17-18,105, 106,107,125, 221,222,231, 305, 311, 363
	9	sort detail	26,27,28, 95-96, 106
	10	bind, commands, and utilities	23-24-25,90, 91,105,106, 107,108-109, 110-111,201, 219,220,256, 360
	11	dispatching	(46,47-48,49- 50,51-52,56- 57,93-94), 106,113
	12	storage manager	(98-99,100-101), 106
	13	edit and validation exits	11,12,19,105, 106,107
	14	in and out of DB2	67,106,121-122
	15	installation-defined performance record	154
	16	events associated with queries to or from other locations	[157,] 158,159, 160-161,162- 163,167,183
	17	drain and claim detail	211, 212, 213- 214, 215-216
	18	DB2 message monitoring	197
	20	data sharing summary	249,250,251, 256,257,261, 262,267-268
	21	data sharing detail	255,259,263, 329
	22	authorization exit information	314
	23	reserved	
	24	stored procedure detail	
	25 -29	reserved	
	30 -32	available for local use	

For a complete list of available IFCID's see section "List of available IFCID's" on page 36.

## 1.2.5 Monitor Traces

The monitor trace records data for online monitoring with user-written programs.

Monitor trace classes 1, 2 and 3 must be active if application programs are to collect IFCID's via the Instrumentation Facility Interface. The Monitor Trace produces many IFCID's that are also produced by other types of traces. Monitor traces are low internal overhead traces. Most of the data is not collected until an application issues a READS IFI call.

Type	Class	Data collected	IFCID's activated
Monitor	1	activates the READS IFCID's	1-2,104,106, 124,129,147, 148,149,150, 199,202,230 232
	2	time in DB2 (CPU and elapsed)	254,306
	3	wait time in DB2 for i/o, locks, latches, drains and claims, asynchronous GBPrequests. Also includes resource usage information.	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, 242-243, 329, 382-383
	4	installation-defined monitor record	155
	5	time spent processing IFI requests	187

Type	Class	Data collected	IFCID's activated
	6	data capture data	185
	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, 242-243, 382-383
	9	SQL-statement-level	124
	10	package detail	239
	11 - 28	reserved	
	29	enhanced monitoring	400,401, 402 (V10)
	30 - 32	Available for local use	

For a complete list of available IFCID's see section "List of available IFCID's" on page 36.

### 1.2.6 Global Traces

The global trace is used by IBM for debugging DB2 itself. The DB2 Diagnostics Guide identifies the types of IFCID's collected by the trace class. Trace class is a parameter on the START TRACE DB2 command that is a short hand way of specifying sets of IFCID's. Typically, the global trace output is directed to an SRV or RES destination. IBM may direct you to start the global trace in response to the reporting of a DB2 problem. You will be given explicit instructions when this is required.

By its nature, global trace can introduce significant overhead into the DB2 system. Global trace classes 1, 2, and especially 3 can significantly degrade system performance and can produce an enormous amount of data if externalized to GTF or SMF.

Type	Class	Data collected	IFCID's activated
Global	1	IBM service	106,(132,134, 138)
	2	IBM service	106,(131,133, 139)
	3	IBM service	0,38,46,47,48, 49,50,51,52,56 57,68,69, 70,71,72,73,74 75,76,77,80, 81,82,83,84, 85,86,87,88,89 93,94,106,114 115,116, 117,174,175, (228-229), (252,260), (265- 266) 267-268, 364
	4	IBM service	106,(130)
	5	IBM service (overflow hybrid join, host variable tracing, DBD invalidation)	190,(135,136, 137),(247-248), 249
	6	user defined service-ability trace	156
	7	IBM service (distributed data) Following events are traced: <ul style="list-style-type: none"> <li>• VTAM exits to DB2</li> <li>• all VTAM calls/returns</li> <li>• Buffer sent/received</li> <li>• Conversation allocation request queued in DB2.</li> </ul>	164,165,166
	8	IBM service (distributed SQL) Following events are traced:	168

Type	Class	Data collected	IFCID's activated
		<ul style="list-style-type: none"> <li>Requesting location SQL statements before modification</li> <li>Both requesting and responding location SQL statements after modification.</li> </ul>	
	9	IBM service (DB2 private protocol and DRDA protocol)	180,181,182
	10	storage manager pool statistics	217
	11	DB2-supplied stored procedure and user- defined function trace	344,345

For a complete list of available IFCID's see section "List of available IFCID's" on page 36.

A readable z/OS data set is shipped with DB2 which describes the DB2 trace codes issued by the global trace facility. The file is called prefix.SDSNSAMP(DSNWEIDS).

For each resource manager, the associated global trace events are identified with event ids (EID's). The originating CSECT, a description of the event, and the type and meaning of the items traced are shown for each EID. Refer to chapter 5-3 of Diagnosis Guide and Reference for a discussion of the global trace.

### 1.3 Trace Overhead

- Statistics Traces: overhead is negligible
- Accounting: 2.5% – 10%
  - class 1,2,3,7,8 are usually affordable
  - class 2 can be expensive for SQL intensive applications (as it traces entry/exit into DB2)
  - class 3 can be expensive with high latch contention (even though no trace record written tracking latch susp/resume, can be expensive if many events/tran)
  - Class 10 (package detail) usually not collected, is more expensive
- Statistics Trace Classes 1,3,4,5 + Accounting Trace Classes 1,3: typically 2% - 5 %
- Audit Trace: < 5%  
With DB2 V8 CPU time measurements IBM observed 7.2% overhead for turning on all audit trace classes for one online transaction. A typical overhead for an online transaction is expected to be less than 10% even when all audit trace classes are turned on. An overhead as high as 10% or more is possible for transactions that execute many short-running and distinct SQL statements just once per transaction, because an audit trace overhead is encountered only the first time an audited table is accessed or updated in a transaction.  
For utility, batch, and long-running SQL, the overhead is practically zero.
- Performance Trace: typically from 5% - 30 %, but up to 100% possible
- Global Trace: 2% – 100%

#### 1.3.1 Minimize Trace Overhead

Using the DB2 trace facility, can consume a large amount of processing resources. Performance trace and global trace are especially resource intensive. Suppressing other trace options, such as TSO, IRLM, z/OS, IMS, CICS, and other trace options, can also reduce overhead. By suppressing DB2 trace options, you might significantly reduce processing costs.

To minimize the amount of processor that is consumed by the DB2 trace facility:

- Enable only the minimal trace and audit classes that you need. You can enable more detailed traces only when you encounter specific performance problems.

- Turn off global trace to significantly reduce processor consumption. Global trace requires 2% to 100% additional processor utilization. If conditions permit at your site, the DB2 global trace should be turned off. To turn off global trace:
  1. Specify NO for the field TRACE AUTO START on panel DSNTIPN at installation.
  2. If the global trace becomes needed for serviceability, use START TRACE command to start it.
- Avoid using accounting class 2 if possible.

Enabling accounting class 2 along with accounting classes 1 and 3 provides additional detail that relates directly to the accounting record IFCID 0003, and records thread level entry into and exit from DB2. Doing so allows you to separate DB2 times from application times.

Running accounting class 2 adds to the cost of processing. How much overhead occurs depends on how much SQL the application issues. Typically, an online transaction incurs an additional 2.5% when running with accounting class 2. A typical batch query application, which accesses DB2 more often, incurs about 10% overhead when running with accounting class 2. If most of your work is through CICS, you most likely do not need to run with class 2, because the class 1 and class 2 times are very close.

However, if you use CICS Transaction Server for z/OS with the Open Transaction Environment (OTE), you should activate and run class 2. If you are concerned about high volume DB2 accounting records, for the DDF or DRDA threads and RRS attach threads, you can reduce the number of DB2 accounting records by using the DB2 system parameter ACCUMACC, which consolidates multiple accounting records into one.

- Consider the overhead of audit trace. When the audit trace is active, the more tables that are audited and the more transactions that access them, the greater the performance impact. The overhead of audit trace is typically less than 5%.

When estimating the performance impact of the audit trace, consider the frequency of certain events. For example, security violations are not as frequent as table accesses. The frequency of utility runs is likely to be measured in executions per day. Alternatively, authorization changes can be numerous in a transaction environment.

- Turn on performance trace classes only for specific performance problems. The combined overhead of all performance classes runs from about 20% to 100%. The overhead for performance trace classes 1 through 3 is typically in the range of 5% to 30%. Consequently, you should turn on only the performance trace classes that are required to address a specific performance problem and qualify the trace as much as possible to limit the that is data gathered to only the data that you need. For example, you qualify the trace by the plan name and IFCID.

You can also qualify traces by any of the following attributes:

- Current or original authorization ID
  - Connection name
  - Correlation name
  - Cached statement ID
  - Location name
  - Workstation ID
  - Package name, collection, or program
  - Transaction or application
  - Role
- Use the STATIME subsystem parameter to control the interval for writing IFCID 105 and 106 records from statistics class 1. Starting statistics traces class 1,3,4, (and 5 for data sharing environments) provides data at the system level. Because statistics trace information is written only periodically, CPU overhead is negligible.

## 1.4 Controlling Traces

Use DB2 Command interface to control DB2 traces:

```
//DSN      EXEC  PGM=IKJEFT01
//STEPLIB  DD    DSN=hilvl.DSNDSLOAD,DISP=SHR
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
DSN S(ssid)
  -START|STOP|MODIFY TRACE(...) ...
END
/*
```

If DB2 is stopped and started after you have started a trace, the trace is not restarted automatically. How DB2 handles DB2 trace depends on installation options on panel “DSNTIPN INSTALL DB2 - TRACING PARAMETERS”:

- **AUDITST(NO|YES|/list\*)**: The AUDITST parameter controls whether the audit trace is to start automatically when DB2 is started. You can specify the classes for which the audit trace is to start.  
NO: Specifies that there is to be no automatic start of the audit trace. If the audit trace is to be used, it must be started with the START TRACE command.  
YES: Specifies that the audit trace for the default class (class 1) starts whenever DB2 is started.  
/list: Specifies that the audit trace for the specified classes starts whenever DB2 is started. To specify classes, list the numbers (any integer from 1 to 32) separated by commas.  
\*: Specifies that the audit trace starts for all classes whenever DB2 is started.
- **TRACSTR(NO|YES|/list\*)**: The TRACSTR subsystem parameter specifies whether the global trace is to start automatically when DB2 is started. This parameter also specifies the classes for which the global trace is to automatically start.  
NO: Specifies no automatic start of the global trace. If the global trace is to be used, it must be started with a special START TRACE command.  
YES: Starts the global trace for the default classes (classes 1, 2, and 3) whenever DB2 is started, and it performs additional data consistency checks whenever a data or index is modified.  
/list: Starts the global trace for the specified classes. Enter a list of class numbers (any integer from 1 to 32) separated by commas. Only classes 1 to 9 are defined by DB2.  
\*: Starts global trace for all classes.
- **SMFACCT(NO|YES|/list\*)**: The SMFACCT subsystem parameter specifies whether DB2 is to send accounting data to SMF automatically when DB2 is started. This parameter also specifies which classes are to be sent. (Default is 1).  
NO: Specifies no automatic start of classes.  
YES: Starts the trace for the default class (class 1). You might also need to update the SMFPRMxx member of SYS1.PARMLIB to permit SMF to write the records.  
/list: Starts the specified classes. Enter a list of class numbers (any integer from 1 to 32), separated by commas. Only classes 1 to 5, 7, and 8 are defined by DB2.  
\*: Starts all classes.
- **SMFSTAT (NO|YES|/list\*)**: The SMFSTAT subsystem parameter specifies whether DB2 is to send statistical data to SMF automatically when DB2 is started. This parameter also specifies which classes are sent.  
NO: Specifies no automatic start.  
YES: Starts the trace for the default classes (classes 1, 3, 4, 5, and 6). You might also need to update the SMFPRMxx member of SYS1.PARMLIB to permit SMF to write the records.  
/list: Starts the specified classes. Enter a list of class numbers (any integer from 1 to 32), separated by commas. Only classes 1 to 5 are defined by DB2.  
\*: Starts all classes.

For a complete list of DB2 trace command options, see DB2 Command Reference Guide.

Most of the constraint options, mentioned in sections below, support the “\*” wildcard and positional “\_” wildcard. Both wildcard symbols may be used in combination to support complex trace situations.

**DEST:** Specified where trace output is to be recorded. If the specified destination is not active or becomes inactive after you issue START TRACE command, you receive message DSNW133I, which indicated that the trace data is lost. For more details see “Trace Record destinations” on page 18.

**IFCID(nnn):** Specifies which other IFCID’s (trace events), in addition to those IFCID’s contained in the classes specified in the CLASS option, are to be started. To start only those IFCID’s specified in the IFCID option, use trace classes 30-32. These classes have no predefined IFCID’s and are available for a location to use. If you do not specify the IFCID option, only those IFCID’s contained in the activated trace classes are started. The maximum number of IFCID’s is 156.

#### Using the IFCID keyword

Is used to specify ADDITIONAL IFCIDs on top of the IFCIDs that are part of the trace class that you are starting, for example ...

```
-STA TRA(P) C(1) IFCID(21)
```

This will trace IFCID 001,002, 031,042,043,076,077,078,079,102,103, 105,107,106,153 (that make up PERF class 1) and also IFCID 21

```
-STA TRA(P) C(1) IFCID(42,43)
```

... will still trace all the IFCIDs that are part of PERF class 1. Specifying IFCID(42,43) is not needed as they are already part of PERF class 1.

If you only want to trace certain IFCIDs, use an “empty” trace class. That is a class that has no IFCIDs assigned to it by default. For the performance trace, these are classes 29, 30, 31, 32. Example: To only trace IFCID 42,43 (system checkpoint start/end) ,use ...

```
- STA TRA(P) C(32) IFCID(42,43)
```

**RMID:** For a list of possible RMID’s, see page 27.

**TDATA:** Specifies the product section headers to be placed into the product section of each trace record. The product section of a trace record can contain multiple headers. All IFC records have a standard IFC header.

**COR:** Places a correlation header on the record.

**TRA:** Places a trace header on the record.

**CPU:** Places a CPU header on the record. The CPU header contains the current processor time for the z/OS TCB or SRB executing.

**DIST:** Places a distributed header on the record.

New with DB2 Version 10:

**STMT:** Places a statement header on the record.

**AUDITPLCY(pol1, poln):** Introduces a list of up to eight audit policy names for which trace information is gathered. AUDITPLCY applies to trace type AUDIT and cannot be specified with CLASS or IFCID.

## 1.4.1 Starting a trace

```
-START TRACE (PERFM|ACCTG|STAT|AUDIT|MONITOR)
  DEST (GTF|SMF|SRV|OPn|OPX)
  {constraint-block}
  {RMID}
  {COMMENT(string)}
  {SCOPE (LOCAL|GROUP)}
```

constraint-block:

```
PLAN() PKGLOC() PKGCOL() PKGPROG() AUTHID() CLASS() TNO()
LOCATION() USERID() APPNAME() WRKSTN() CONNID() CORRID()
filter-block IFCID() BUFSIZE() TDATA(COR|TRA|CPU|DIST)
```

```
IFC exclude filtering:  
XPLAN() XPKGLOC() XPKGCOL() XPKGPROG() XAUTHID() ... etc
```

## 1.4.2 Stopping a trace

```
-STOP TRACE (PERFM|ACCTG|STAT|AUDIT|MONITOR)  
DEST (GTF|SMF|SRV|OPn|OPX)  
{constraint-block}  
{RMID}  
{COMMENT(string)}  
{SCOPE (LOCAL|GROUP)}
```

## 1.4.3 Modifying a trace

```
-MODIFY TRACE (PERFM|ACCTG|STAT|AUDIT|MONITOR)  
CLASS() TNO() IFCID()  
{COMMENT(string)}  
{SCOPE (LOCAL|GROUP)}
```

## 1.4.4 Displaying a trace

```
-DISPLAY TRACE (PERFM|ACCTG|STAT|AUDIT|MONITOR)  
DEST (GTF|SMF|SRV|OPn|OPX)  
{constraint-block}  
{DETAIL(output-type)}  
{COMMENT(string)}  
{SCOPE (LOCAL|GROUP)}  
{RMID}
```

DETAIL: Limits the information that a trace displays based on the output type specified within parentheses. The possible values for output-type are:

- 1 Display summary trace information: TRACE NUMBER, TYPE, CLASS, DEST
- 2 Display qualification trace information: TRACE NUMBER, AUTHID, LOCATION, PACKAGE, PLAN, TXACT, USERID, WORKSTATION
- 1,2 Display both summary and qualification information
- \* Display both summary and qualification information

## 1.4.5 DSN1SDMP – selective dumps

DSN1SDMP is a IBM DB2 batch utility which starts a and then watches the IFCID records being produced. It checks for certain conditions in those records and takes an action when those conditions are met. Then stops itself. It is most used for taking a dump on the occurrence of certain SQLCODE.

You must specify the OPX destination for all traces that are being recorded to an OPn buffer for the DSN1SDMP utility to use. By specifying the OPX destination, you avoid the possibility of starting a trace to a buffer that is already assigned.

For more details and a complete description, see DB2 for z/OS Utility Guide and Reference.

**DSN1SDMP example 1: Catch SQLCODE -301 (= x'FFFFFFED3')**

```
//SDMPIN DD *  
START TRACE=P CLASS(32) IFCID(340) DEST(OPX) TDATA(TRA)  
AFTER(1)  
FOR(1)  
ACTION(ABENDTER(00E60189))  
SELECT
```

```
P4,00
DR,04,X'0154'
P4,08
DR,16,X'FFFFFFED3'
/*
```

#### DSN1SDMP example 2: Catch SQLCODE -904 (= x'FFFFFFC78')

```
//SDMPIN DD *
START TRACE=A CLASS(32) IFCID(53,58) DEST(OPX)
FOR(1)
AFTER(1)
ACTION(ABENDRET(00E60188))
SELECT
* Position to the prodcut section
P4,00
* Insure QWHSIID = 58 or 53 (not IFCID 4)
GE,04,X'0005'
* Position to the data section 1
P4,08
* Compare SQLCODE in QW0058SQ or QW0053SQ
DR,74,X'FFFFFFC78'
/*
```

#### DSN1SDMP example 3: Catch EDMPOOL FULL Condition (= x'00C90089')

```
//SDMPIN DD *
* Note: This DSN1SDMP will cause abend RET (retry) on an EDM pool
* full condition. It is important that both MSTR and DBM1
* address spaces are in the dump and that the dump is complete.
* Verify this before sending in the dump.
* QWHS OFFSETS
* 0 LEN 2
* 2 TYPE 1
* 3 RMID 1
* 4 IFCID 2
* QWHT OFFSETS
* 0 LEN 2
* 2 TYPE 1
* 4 EID 2
*
*
START TRACE=P CLASS(30) DEST(OPX) IFCID(31) TDATA(COR,TRA)
BUFSIZE(8)
AFTER(1)
FOR(1)
SELECT
* POSITION TO HEADERS (QWHS IS ALWAYS FIRST)
P4,00
* CHECK QWHS 01, FOR RMID 14, IFCID 31, EDM pool full condition
DR,02,X'010E001F'
ACTION(ABENDRET(00E60107))
/*
```

#### DSN1SDMP example 4: Catch SQLCODE -805 (= x'FFFFFFCDB')

```
//SDMPIN DD *
START TRACE=A CLASS(32) IFCID(53,58) DEST(OPX)
FOR(1)
```

```

AFTER(1)
ACTION(ABENDRET(00E60188))
SELECT
* Position to the product section
P4,00
* Insure QWHSIID = 58 or 53 (not IFCID 4)
GE,04,X'0005'
* Position to the data section 1
P4,08
* Compare SQLCODE in QW0058SQ or QW0053SQ
DR,74,X'FFFFFFCDB'
/*

```

#### Required JCL:

```

//IFCSD EXEC PGM=IKJEFT01
//STEPLIB DD DISP=SHR,DSN=prefix.SDSNLOAD
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SDMPRNT DD SYSOUT=*
//SDMPTRAC DD DISP=(NEW,CATLG,CATLG),DSN=IFCSD.TRACE,
// UNIT=SYSDA,SPACE=(8192,(100,100)),DCB=(DSORG=PS,
// LRECL=32756,RECFM=VB,BLKSIZE=32760)
//SDMPIN DD *
as described in above examples
/*
//*****
****
//SYSUDUMP DD SYSOUT=*
//SYSTSIN DD *
DSN SYSTEM(DSN)
RUN PROG(DSN1SDMP) [PLAN(DSNEDCL)]
END
/*

```

#### Output:

Trace records that DB2 writes to SDMPTRAC are of the same format as SMF or GTF records except that the SDMPTRAC trace record headers contain the monitor header (that is mapped by DSNDQWIW, see “IFCID mappings” on page 38). The The DCB parameters are VB, BLKSIZE=32760, LRECL=32756.

Using ISPF/BROWSE or IPCS you may browse the SDMPTRAC dataset.

For information about format of trace record see section “Format of Trace Records” on page 29.

#### End / Abend:

Abend reason codes 00E60100 through 00E60199 indicate that you have requested a dump using the DSN1DMP utility.

If DSN1SDMP does not finish execution, you can stop the utility by issuing the STOP TRACE command, as in the following example:

```
-STOP TRACE=P CLASS(32)
```

### 1.4.6 DB2 Utilities with DIAGNOSE

For utility jobs with a performance problem, you can add DIAGNOSE TYPE(100,101,102) to the job and compare the job outputs and trace data.

TYPE(integer, ...) Specifies one or more types of diagnose that you want to perform. integer is the number of types of diagnoses. The maximum number of types is 32.

### Example:

```
//DB2UTIL EXEC DSNUPROC,UID='MYID.REBUILD',
//          UTPROC=' ',SYSTEM='SSTR'
//SYSIN DD *
DIAGNOSE TYPE(100,101,202)
          REBUILD INDEX (IDOS0302, IDOS0304, IPOS0301)
          SORTDEVT SYSDA
DIAGNOSE END
/*
```

## 1.4.7 Authorization

To execute DB2 TRACE commands, you must use a privilege set of the process that includes one of the following authorities or privilege:

- SYSOPR authority
- SYSCTRL authority
- SYSADM authority
- or system privilege which is included in above mentioned authorities:
  - TRACE: Grants the privilege to issue the MODIFY TRACE, START TRACE, and STOP TRACE commands

On the first READA or READS call from a user, an authorization is checked to determine if the primary authorization ID or one of the secondary authorization IDs of the plan executor has MONITOR1 or MONITOR2 privilege:

- MONITOR1: Grants the privilege to obtain IFC data classified as serviceability data, statistics, accounting, and other performance data that does not contain potentially secure data.
- MONITOR2: Grants the privilege to obtain IFC data classified as containing potentially sensitive data such as SQL statement text and audit data. Users with MONITOR2 privileges have MONITOR1 privileges.

If you have an authorization failure, an audit trace (class 1) record is generated that contains the return and reason codes from the exit. This is included in IFCID 0140.

DB2 commands that are issued from a logged-on z/OS console or TSO SDSF can be checked by DB2 authorization using primary and secondary authorization IDs.

## 1.5 Trace Record destinations

Trace records can be written to SMF, GTF, or a DB2 online performance (OP) buffer. Regardless of whether you write records to SMF, GTF, or OP, each consists of four basic parts:

1. an SMF, GTF, or OP writer header section
2. a self-defining section
3. a product section
4. zero or more data sections.

The length and content of the writer header section differs between smf records, gtf records, and op records. Sections of the records are the same for SMF, GTF and OP. For more information, see "programming for the instrumentation facility interface" in the information management software for z/OS solutions information center.

### **Allowable and default destinations for each trace type:**

Type	GTF	SMF	SRV	OP <sub>n</sub>	OPX
PERFM	default	allowed	allowed	allowed	allowed
ACCTG	allowed	default	allowed	allowed	allowed

		SMF 101			
<b>STAT</b>	allowed	default SMF 100 + 102	allowed	allowed	allowed
<b>AUDIT</b>	allowed	default SMF 102	allowed	allowed	allowed
<b>MONITOR</b>	allowed	allowed	allowed	allowed	default

### Trace destination details:

- GTF: The z/OS generalized trace facility (GTF). The record identifier for records from DB2 is X'0FB9'.
- SMF: The system management facility. The SMF record type of DB2 trace records depends on the IFCID record, as follows:

IFCID record: → SMF record type:

1 (System Services Statistics) → 100

2 (Database Services Statistics) → 100

3 (Agent Accounting) → 101

202 (Dynamic System Parameters) → 100

225 (System Storage® Summary Statistics) → 100

230 (Data Sharing Global Statistics) → 100

239 (AGENT ACCOUNTING OVERFLOW) → 101

All Others (e.g. of a performance trace) → 102

Generally spoken: Trace statistics data are in SMF record type 100, accounting data in 101 and other trace data in SMF type 102 records.

- SRV: An exit to a user-written routine. For instructions and an example of how to write such a routine, see the macro DSNWVSER in library prefix .SDSNMACS.
- OPn: A specific destination. n can be an integer from 1 to 8.
- OPX: A generic destination which uses the first free OP n slot. Only applications that start a trace to an OPn buffer can read that buffer.

The maximum length of an SMF record is 32,756 bytes; any records that exceed this length are truncated before they are written to the SMF data set.

GTF records are blocked to 256 bytes. Because some of the trace records exceed the GTF limit of 256 bytes, they have been blocked by DB2.

DB2 data should usually be directed to SMF instead of GTF for several reasons. First and most important, the "write" of DB2 data to GTF requires an actual physical I/O, managed by DB2, which can interfere with performance measurements. The "write" of DB2 data to SMF is not a physical write, but is a data movement at memory speed from the DB2 address space to the SMF address space, which increases the accuracy of DB2 timestamps. Second, SMF records are 32760 bytes long, but GTF is limited to 256 bytes in each physical record, which means that not only is the overhead of creation significantly less with SMF than with GTF, but also your processing programs will be more efficient using SMF data. Finally, if the logical data length exceeds 256 bytes to GTF, the spanning format is used, in which actual data fields can be split across records.

### Special Accounting Trace Feature

Requesting accounting trace data is verify special in that, starting additional trace classes don't create extra trace records, they just add counters to existing records. The result of that is that if ...

```
-STA TRA(A) C(1) DEST(SMF)
-STA TRA(A) C(1,2,3) DEST(OPX)
```

the SMF records will also contain the class 2 and 3 data. The record is only built once in storage.

### **SMF Record Subtypes:**

DB2 SMF records will have the SMFxF LG set to indicate a subtype is valid.

SMF Record Type	Description
Type 70 (Subtype 1)	RMF processor activity statistics
Type 102 (Subtype 106)	ZPARMS information - optional
Type 102 (Subtype 172)	Deadlock trace records - optional
Type 102 (Subtype 196)	Timeout trace records - optional
Type 102 (Subtype 225)	DBM1 storage trace records - optional

## **1.6 RMID's**

Each IFCID is associated with a specific RMID - resource manager identifier.

1	for initialization procedures	20	service controller
2	agent services management	21	database utilities
3	recovery management	22	relational data systems
4	recovery log management	23	general command processing
6	storage management	24	message generator
7	subsystem support for allied memories	25	distributed relational data system
8	subsystem support for SSI functions	26	instrumentation accounting and statistics
10	buffer management	27	data communications resource manager
12	system parameter management	28	distributed transaction manager
14	data manager	29	distributed data interchange service
16	instrumentation cmd's, trace, dump services	30	distributed transaction manager (DSCF address space)
18	data space management	31	group manager

A readable z/OS data set is shipped with DB2 which describes the DB2 trace codes issued by the global trace facility. The file is called prefix.SDSNSAMP(DSNWEIDS).

For each resource manager, the associated global trace events are identified with event ids (EID's). The originating CSECT, a description of the event, and the type and meaning of the items traced are shown for each EID. Refer to chapter 5-3 of Diagnosis Guide and Reference for a discussion of the global trace.

## **1.7 How it works internally**

To use the READA call, the application's start trace command must specify a special parameter area address, and the size of a destination buffer that DB2 uses to temporarily hold the trace data until the application calls for it. The parameter area contains a percent value and the address of an event control block (ECB) in the applications addressable storage upon which the application will eventually wait. The destination buffer is called an OP buffer, and is allocated in the extended common storage area. When the OP buffer becomes full to the specified percentage, DB2 will POST the ECB, thereby notifying the application that there is data to be retrieved.

The application responds to the POST by first clearing the ECB and then issuing a READA IFI call. When the application makes it's READA call, it must supply the address of a local buffer to receive the collected data. The Instrumentation Facility will move the IFCID's from the OP buffer to the local buffer specified in the READA call. When control is returned to the application the READA IFCID's are available for application processing. The READA caller is not suspended while DB2 collects the requested data, but is suspended while the collected IFCID's are moved to its local buffer.

IFI allocates up to eight OP buffers upon request from private storage in the DB2 MSTR address space. IFI uses these buffers to store trace data until the owning application performs a READA request to transfer the data from the OP buffer to the application's return area. An application becomes the owner of an OP buffer when it issues a START TRACE command and specifies a

destination of OPn or OPX. Each buffer can be of size 256 KB to 16 MB. IFI allocates a maximum of 16 MB of storage for each of the eight OP buffers. The default monitor buffer size is determined by the MONSIZE parameter in the DSNZPARM module.

The start trace command specifies the IFCID destination either by number (i.e. OP1), or generically with the OPX designation. OPX will use the next available OP number. OP buffers are filled by a wrap-around technique. If the application does not call for the data soon enough, DB2 may overwrite the OP buffer. In this case a console message is issued by the Instrumentation Facility Resource manager indicating that data has been lost.

The space allocated for the OP buffer is obtained under the TCB of the application that issues the start trace command. When the space for the buffers is accounted for, it will appear to belong to the application that started the trace, not to DB2.

OP buffer space is not released when the trace that caused it's allocation ends. DB2 will reuse that space for subsequent trace requests if possible. If a second trace asks for a buffer larger than that used by the first trace, DB2 will allocate the larger buffer size, ignoring the buffer used by the first trace. This process will continue up to the four megabyte limit on OP buffer space, at which point DB2 will refuse to start subsequent traces.

The OPX destination specification should be used with caution. If two distinct applications issue identical start trace commands using OPX, the second trace will not be started. DB2 will issue a message stating that the trace is already started. This may cause the application issuing the start trace command to fail.

The READS call is a synchronous request for data. A caller supplies a buffer to hold requested data, and is suspended until the IFI has moved the required IFCID's into the buffer. No OP buffer is associated with a READS call.

### **Information about ZIIP CPU times**

All ZIIP CPU times that are reported in DB2 trace records are normalized to the speed of the purpose Processor. ZIIP CPU times that are reported by RMF are not normalized. If you need to compare the ZIIP CPU times that are reported by DB2 to the times that are reported by RMF, you need to use the normalization factor that is specified by the r723nffs field in the workload manager control section of the SMF type 72 record: RMF workload activity and storage data. see the z/OS MVS system management facilities manual for details about normalization.

## **1.8 DSN1SMFP - printing SMF records**

APAR PK34261 added DSN1SMFP in 2007 the first time to DB2 V8. DSN1SMFP The PTF to the APAR introduced the DB2-supplied application for processing DB2 trace data into reports that are useful for evaluating and auditing the DB2 environment for Common Criteria. It also adds a sample job called *prefix.SDSNSAMP(DSNTEJCC)* that shows how to call DSN1SMFP.

DSN1SMFP accepts data that SMF collects in standard SMF format and produces from one to eighteen reports. DSN1SMFP accepts all SMF record types, but it processes only type 101 (DB2® Accounting) and 102 (DB2 Performance) records. DSN1SMFP checks each type 101 and 102 record for DB2 audit trace types of these DB2 IFCIDs:

- |  |  |                            |
|--|--|----------------------------|
| • 003: Accounting - DDF Data by Location (security-relevant fields only) | • 106: System Parameters (security-relevant fields only) | • 144: Audit First Read    |
| • 004: Trace Start   | • 140: Audit Authorization Failures                      | • 145: Audit DML Statement |
| • 005: Trace Stop  | • 141: Audit DDL Grant/Revoke                            | • 269: Trusted Connection  |
| • 023: Utility Start   | • 142: Audit DDL Create/Alter/Drop                       | • 270: Trusted Context     |
| • 024: Utility Change  | • 143: Audit First Write                                 | • 350: SQL Statement       |
| • 025: Utility End   |  |                            |
| • 083: Identify End  |  |                            |

Each such trace type is extracted and outputted in report format to a dedicated DD for that trace type.

### **Sample JCL:**

```
//DSN1SMFP EXEC PGM=DSN1SMFP,COND=(4,LT)
```

```

//STEPLIB DD DISP=SHR,DSN=prefix.SDSNEXIT <- req'd for local
DSNHDECP
// DD DISP=SHR,DSN=prefix.SDSNLOAD
//SMFIN DD DISP=SHR,DSN=SMF records with DB2 trace data
//SYSPRINT DD SYSOUT=* <- messages and EOJ
summary
//IFCID003 DD ...
//IFCID004 DD ...
//IFCID005 DD ...
...

```

IBM first removed DSN1SMFP from SDSNLOAD with DB2 for z/OS Version 10 but then reintroduced the program first with DB2 11 for z/OS and in 2014 also in DB2 10 for z/OS.

## 1.9 DSNTSMFD - SMF Decompression

In DB2 Version 10 for z/OS, the subsystem parameter SMFCOMP can be used to turn on compression of DB2 SMF records. The default value for SMFCOMP is OFF.

The PTF PM27872/UK64597 (Nov. 2011) adds DSNTSMFD to V10. DSNTSMFD is a DB2-supplied sample application for decompressing compressed DB2 SMF records. It accepts an SMF data set in standard SMF format and produces an output SMF data set. All SMF record types are accepted, but only compressed type 100 (DB2 Statistics), type 101 (DB2 Accounting), type 102 (DB2 Performance) records generated by DB2 are decompressed. All other records (including uncompressed DB2 records) are copied unchanged to the output SMF data set.

### Capabilities of DSNTSMFD:

- DSNTSMFD can process input SMF records that contain records from up to 512 DB2 subsystems.
- DSNTSMFD accepts SMF records from any release of DB2. Only systems with the z/OS CSCESRV macro available will be able to use DSNTSMFD to decompress records. Only compressed SMF records from DB2 V10 and beyond are eligible for decompression.
- DSNTSMFD produces an end-of-job summary, which is written to the SYSPRINT DD:
  - Total SMF records read
  - Total DB2 records read
  - Total DB2 compressed records read
  - Total DB2 compressed records decompressed
  - Total non-DB2 records read
  - Aggregate size of all input records
  - Aggregate size of all input DB2 records
  - Aggregate size of all input DB2 compressed records
  - Aggregate size of all output DB2 records
  - Aggregate size of all DB2 expanded records
  - Aggregate size of all non-DB2 input records
  - Percentage saved using compression
  - Details of records by DB2 subsystem

The SYSPRINT DD is also the destination for diagnostic and warning messages generated by DSNTSMFD.

- DSNTSMFD requires the following DD statements:

- SMFINDD (input)  
The SMFINDD DD must specify one or more data sets that contain DB2 trace records in standard SMF format. All SMF records are acceptable, but DSNTSMFD will only attempt to decompress SMF type 100, 101, and 102 records.
- SMFOUTDD (output)  
The SMFOUT DD specifies the output data set. Non-type 100, 101, and 102 SMF records are written to this data set; if the decompression service (CSRCESTRV) is not available, the compressed records are copied and a warning message is produced.
- SYSPRINT (output)  
Contains the end of job summary report plus diagnostic messages for any DSNTSMFD processing exceptions. All messages are documented in the DB2 for z/OS Messages manual.

#### Execution of DSNTSMFD:

DSNTSMFD is designed for execution in the z/OS batch process. It has no dependency on DB2 other than as a source of the SMF records.

#### Code provided and sample job DSNTSJDS

Code is provided by IBM as member DSNTSMFD of library SDSNSAMP. Code is written in assembler. Users need to compile and link the code into the RUNLIB.LOAD library.

Sample job DSNTSJDS provides the basic JCL framework needed to assemble and execute DSNTSMFD.

### **1.10 All traces, classes and IFCID's**

There are more than 280 different IFCID's. Some internal events are of significant duration, and require the production of a pair of IFCID's, one representing the beginning of the event, the other the end. Such pairs are useful for the computation of elapsed times. Less lengthy events are represented by a single IFCID.

While the physical components of an IFCID are various and sundry, for practical purposes there are only two parts to an IFCID: the product section and the data section.

The product section identifies that which is being traced. The start trace command allows the specification of special "headers" that will be included by IFI when the IFCID is produced. These headers contain data on cpu time consumption, correlation information for identification purposes, and information about the use of Distributed Data Facilities (DDF).

The correlation header contains fields which identify both the consumer and the provider of resources. Some of these fields are the Resource Manager ID of the resource provider, an Agent Control Element address of the consumer of resources, and the Authorization ID of the user associated with the resource consumption.

Cumulative CPU consumption of an execution unit (TCB or SRB) is recorded in the CPU headers contained within the Product section of the IFCID if so requested on the START TRACE command.

The purpose of the data section is to record specific information key to understanding the internal DB2 operation represented by the IFCID. Some data section fields are characterized as "serviceability" fields by IBM, and available to the user at his own risk. No further information about such fields can be expected from IBM. These fields appear to contain DB2 internal debugging data, and are typically only used under the direction of IBM. Most IFCID's have data fields that describe events which are pertinent to the performance tuning and problem analysis processes.

Some IFCID's have individual significance. The IFCID 031 for example represents the EDM Pool Full condition. When space is exhausted in the Environmental Descriptor Management storage pool, this IFCID will be produced by the Data manager to record the occurrence, provided some trace was started that specified that this IFCID be collected.

Another example is the IFCID 024, Utility change of phase. IFCID's such as these are useful for identifying events that may be problems, or as in the case of the IFCID 024, just noteworthy.

Each SQL statement has a beginning IFCID (59,60,61,... ) and an end IFCID (058) that can be used to calculate the duration and cpu resource consumption of the SQL statement. Further the 058 record carries the SQLCA with the result of the SQL call.

CREATE THREAD and TERMINATE THREAD elapsed times can be identified by the 072-073 and 074-075 pairs. In fact, even events such as storage requests can produce IFCID's if requested to do so by a TRACE command. Care must be exercised so that too many IFCID's are not generated by the trace request. Collecting storage request IFCID's may be necessary sometimes, but should be done judiciously, since the overhead of such traces is enormous.

Some IFCID's have no data sections at all. They are markers in time. IFCID 074 (Begin Terminate Thread) has no data field, neither do the 042-043 begin-end Checkpoint IFCID's.

The remainder of the IFCID's contain reference data about the DB2 environment. IFCID's 104, 105, 106, and 107, as well as IFCID's 001, 002, and 003 contain environmental and/or summary data. These IFCID's can give a system wide view of DB2.

Type	Class	Data collected	IFCID's activated
not specific	30 – 32	IFCID's not prepared for use in distinct classes	free choice
not specific	-	available only through an IFI READS call: Location and name of the DSNHDECP module at DB2 startup.	373
Statistics	1	statistical data	1-2,105,106, 202,225
	2	installation-defined statistics record	152
	3	deadlock and lock timeout information, connect or disconnect from a group buffer pool, long-running UR's	172,196,250, 261,262,313, 258, 335, 337
	4	DB2 exception conditions	173,191,192, 193,194,195, 203,204,205, 206,207,208, 209,210,235, 236,(238), 267,268,343, 402
	5	data sharing global statistics	230,254
	6	DBM1 storage summary	0225
	7	DB2 V10 statistics about communication with DRDA remote locations	365
	8	buffer pool data set statistics	0199
Accounting	1	accounting data	3,106
	2	in DB2 time	232
	3	wait time for i/o, locks, latches, drains, and claims, asynchronous GBP requests	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, 242-243, 329, 351, 352, 378, 379
	4	installation-defined accounting record	151
	5	time spent processing IFI requests	187

Type	Class	Data collected	IFCID's activated
	7	package level accounting in-DB2 time	232,239,240
	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, 239,241-243, 351, 352, 378, 379
	10	package detail	339
Audit	1	authorization failures	140
	2	explicit grant and revoke	141
	3	create, drop, and alter operations against audit tables	142
	4	first change of audited object	143
	5	first read of audited object	144
	6	SQL statement at bind	145
	7	change in authorization	55, 83, 87,169, 319
	8	utility access to any object	23, 24, 25, 219, 220
	9	installation-defined audit record	146
	10	trusted context audit	269, 270
	11	DB2 V10 extended authorization checking	271, 361, 362
Monitor	1	activates the reads IFCID's	1-2,104,106, 124,129,147, 148,149,150, 199,202,230 232
	2	time in DB2 (CPU and elapsed)	254,306
	3	wait time in DB2 for i/o, locks, latches, drains and claims, asynchronous GBP requests. Also includes resource usage information.	6-7,8-9,32-33, 44-45,51-52, 56-57,117-118, 127-128,170-171,174-175, 213-214,215-216,226-227 242-243, 329, 378, 379
	4	installation-defined monitor record	155
	5	time spent processing IFI requests	187
	6	data capture data	185
	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, 242-243, 378, 379
	9	SQL-statement-level	124
	10	package detail	339
29	DB2 V10 enhanced monitoring	316, 318, 400, 401	
Performance	1	background event	1-2,31,42-43, 76-77,78-79, 102,103,105, 106,107,153
	2	subsystem related events	3,68-69, 70-71,72-73, 74-75,80-81, 82-83,84-85, 86-87,88-89, 106,174-175, 321,322

Type	Class	Data collected	IFCID's activated
	3	SQL-related events	173,22,53,55, 58,59,60,61, 62,63,64,65, 66,92,95-96, 97,106,112, 177,233,237, 272,311,324, 343, 360
	4	buffer manager i/o and EDM pool requests	6-7,8-9-10, 29-30,105, 106,107,127- 128, 226-227, 357, 358, 359
	5	log manager	32-33,34-35, 36-37,38-39, 40-41,104,106 114-115-116, 117-118,119- 120,(228-229)
	6	summary lock information	20,44-45, 105,106,107, 172,196,213- 214,218, 337
	7	detailed lock information	21,105,106, 107,223
	8	data manager detail	13-14,15-16- 17-18,105, 106,107,125, 221,222,231, 305, 311, 363
	9	sort detail	26,27,28, 95-96, 106
	10	bind, commands, and utilities	23-24-25,90, 91,105,106, 107,108-109, 110-111,201, 219,220, 256, 360
	11	dispatching	(46,47-48,49- 50,51-52,56- 57,93-94), 106,113
	12	storage manager	(98-99,100-101), 106
	13	edit and validation exits	11,12,19,105, 106,107
	14	in and out of DB2	67,106,121-122
	15	installation-defined performance record	154
	16	events associated with queries to or from other locations	[157,] 158,159, 160-161,162-163,167,183
	17	drain and claim detail	211, 212, 213- 214, 215-216
	18	DB2 message monitoring	197
	19	Trace data set open and close	370, 371
	20	data sharing summary	249,250,251, 256,257,261, 262,267-268
	21	data sharing detail	255,259,263, 329
	22	authorization exit information	314
	24	Stored procedure detail	380, 499
Global	1	IBM service	106,(132,134, 138)
	2	IBM service	106,(131,133, 139)
	3	IBM service	0,38,46,47,48, 49,50,51,52,56 57,68,69, 70,71,72,73,74 75,76,77,80, 81,82,83,84, 85,86,87,88,89 93,94,106,114 115,116, 117,174,175, (228-229), (252,260), (265-266) 267-268, 364
	4	IBM service	106,(130)

Type	Class	Data collected	IFCID's activated
	5	IBM service (overflow hybrid join, host variable tracing, DBD invalidation)	190,(135,136, 137),(247-248), 249
	6	user defined service-ability trace	156
	7	IBM service (distributed data)	164, 165, 166
	8	IBM service (distributed SQL)	168
	9	IBM service (DB2 private protocol and DRDA protocol)	180,181,182
	10	storage manager pool statistics	217
	11	DB2-supplied stored procedure and user-defined function trace	344, 345

A dash between IFCID's means they are related. IFCID'S in parentheses are for serviceability.

IFCID 157 is removed from DB2 for z/OS Version 10.

For a complete list of available IFCID's see section "List of available IFCID's" on page 36.

### 1.11 Loading IFCID description into a table

For an comfortable access to IBM's descriptions of IFCID's and traces you may load content of an DB2 distribution library into DB2 tables. Input comes from member DSNWMSGs in IVP library DSNIVPD.

After creation of 2 tables and loading the data you can use SQL to retrieve the trace field descriptions in whatever format or order you need. You can execute the SQL statements as you would any other SQL (for example, through SPUFI, QMF, or DSNTEP2). Some sample SQL statements follow that might be useful in your analysis of DB2 trace records. Consider printing a formatted report of the results from the first three queries below; they might provide useful reference material for your system performance analyst or auditor.

```
-- TO RETRIEVE ALL TRACE FIELD DESCRIPTIONS, ORDERED NUMERICALLY BY IFCID:
SELECT DISTINCT *
  FROM TB_TRACE_DESCR
  ORDER BY SEQ;

-- TO RETRIEVE ALL TRACE FIELD DESCR., ORDERED ALPHABETIC. BY FIELD NAME:
SELECT DISTINCT *
  FROM TB_TRACE_DESCR
  ORDER BY NAME, SEQ;

-- TO RETRIEVE ALL TRACE FIELD DESCRIPTIONS, ORDERED BY TRACE TYPE AND
-- TRACE CLASS:
SELECT TYPE, CLASS, A.IFCID, DESCRIPTION, SEQ
  FROM TB_TRACE_TYPES A,
       TB_TRACE_DESCR B
  WHERE A.IFCID = B.IFCID
  ORDER BY TYPE, CLASS, SEQ;

-- TO RETRIEVE ONLY THE FIELD DESCRIPTIONS FOR
-- TRACE RECORD IFCID 21:
SELECT DISTINCT *
  FROM TB_TRACE_DESCR
  WHERE NAME = 'COPY' OR IFCID = 21
  ORDER BY SEQ;

-- TO RETRIEVE ONLY THE FIELD DESCRIPTION FOR A FIELD CALLED QWACRINV:
```

```

SELECT DISTINCT *
  FROM TB_TRACE_DESCR
 WHERE NAME = 'COPY' OR NAME = 'QWACRINV'
 ORDER BY SEQ;

-- TO RETRIEVE FIELD DESCRIPTIONS FOR AUDIT TRACE RECORDS:
SELECT DISTINCT NAME, DESCRIPTION, A.IFCID, SEQ
  FROM TB_TRACE_DESCR A,
       TB_TRACE_TYPES B
 WHERE A.IFCID = B.IFCID
       AND (TYPE = 'AUDIT' OR NAME = 'COPY')
 ORDER BY SEQ;

-- TO RETRIEVE FIELD DESCRIPTIONS FOR MONITOR TRACE RECORDS:
SELECT DISTINCT A.NAME, A.DESCRPTION,
  A.IFCID, A.SEQ
  FROM TB_TRACE_DESCR A,
       TB_TRACE_TYPES B
 WHERE A.IFCID = B.IFCID
       AND (TYPE = 'MONITOR' OR NAME = 'COPY')
 ORDER BY SEQ;

-- TO RETRIEVE FIELD DESCRIPTIONS FOR PERFORMANCE
-- TRACE RECORDS, CLASSES 1 THROUGH 7:

SELECT DISTINCT NAME, DESCRIPTION, A.IFCID, SEQ
  FROM TB_TRACE_DESCR A ,
       TB_TRACE_TYPES B
 WHERE A.IFCID = B.IFCID AND
       ((TYPE = 'PERFORMANCE' AND
         CLASS IN (1,2,3,4,5,6,7))
        OR NAME = 'COPY')
 ORDER BY SEQ;

```

To aid the task of analyzing your site's performance reports, consider creating a 3-column table called TABLE\_X, where each row contains a report name, a report field sequence number, and a report field name (as it appears in the name column of TB\_TRACE\_DESCR, not necessarily the same name that appears on the formatted report). Enter a row in TABLE\_X for each field on each report. You then execute SQL statements like the following to create a tailored set of report field descriptions for each report in use at your site.

```

-- TO RETRIEVE FIELD DESCRIPTIONS FROM REPORT A IN THE
-- ORDER THAT THEY APPEAR ON THE REPORT:

SELECT DISTINCT A.NAME, A.DESCRPTION, A.SEQ, B.SEQ
  FROM TB_TRACE_DESCR A,
       TABLE_X B
 WHERE B.REPORT = 'REPORT A' AND
       B.NAME = A.NAME
 ORDER BY B.SEQ, A.SEQ;

```

### 1.11.1 Create Tablespace and Tables

Before loading the data, you need to create tables for the data:

```

CREATE TABLESPACE TSIFCID
  IN DSNDB04
  -- USING STOGROUP DSN8G110

```

```

;
CREATE TABLE TB_TRACE_DESCR
  (IFCID      INTEGER NOT NULL,
   NAME       CHAR(20) NOT NULL,
   DESCRIPTION CHAR(60) NOT NULL,
   SEQ        INTEGER NOT NULL)
  IN DSNDB04.TSIFCID
;
CREATE TABLE TB_TRACE_TYPES
  (TYPE       CHAR(17) NOT NULL,
   CLASS      INTEGER NOT NULL,
   IFCID      INTEGER NOT NULL,
   COMMENTS   CHAR(45) NOT NULL WITH DEFAULT)
  IN DSNDB04.TSIFCID
;

```

## 1.11.2 Loading IFCID descriptions and Trace types

After successful creation of the 2 tables, you can now load the data:

```

//LOAD EXEC DSNUPROC,SYSTEM=ssid,
// LIB='SYS1.DSNVnnn.SDSNLOAD'
//DSNUPROC.SYSREC DD DISP=SHR,DSN=SYS1.DSNVnnn.SDSNIVPD(DSNWMSG)
//DSNUPROC.SYSIN DD *
    LOAD DATA INDDN(SYSREC)
        LOG(NO)
        REPLACE
        STATISTICS TABLE(ALL) INDEX(ALL)
    INTO TABLE TB_TRACE_DESCR WHEN (1) = '0'
        (IFCID      POSITION(1:4) INTEGER EXTERNAL(4),
         NAME       POSITION(6:25) CHAR(20),
         DESCRIPTION POSITION(27:86) CHAR(60),
         SEQ        POSITION(87:92) INTEGER EXTERNAL(6) )
    INTO TABLE TB_TRACE_TYPES WHEN (1) = '1'
        (TYPE       POSITION(3:19) CHAR(17),
         CLASS      POSITION(27:28) INTEGER EXTERNAL(2),
         IFCID      POSITION(35:39) INTEGER EXTERNAL(5),
         COMMENTS   POSITION(42:86) CHAR(45) )

-- ADD COPY option to above LOAD statement or
-- execute COPY utility afterwards!
/*

```

## 1.12 Tools processing DB2 Trace data

### 1.12.1 Format of Trace Records

Format of the trace records:

- **Header:** The header of the trace record depends on the destination type and could contain the SMF header, the IFI header, or the GTF header based on the destination specified in the start trace command.
- **Self-defining section:** This section resides next to the header and contains a pointer to the product section followed by as many pointers as data sections. The pointer contains triplet fields – offset of the section, length of the section, and number of times the section is repeated.
- **Data section:** One or more data sections follow the self-defining section and they have to be tracked down using pointers provided in the self-defining section. These data sections contain the actual trace data, requested through the START TRACE command.

- Product section: The last section of the trace record is the product section and the pointer provided in the self-defining section tracks its position. The product section for all record types contains the standard header. The standard header contains information like IFCID, number of data sections (self-defining sections), Subsystem-id, etc. Other headers – correlation, CPU, distributed, and data sharing data – may also be present. The information in the product section header is controlled through TDATA options of the START TRACE command. TDATA specifies the product section headers to be placed into the product section of each trace record. If you do not specify TDATA, then the type of trace determines the type of product section header. The product section of a trace record can contain multiple headers.

All IFC records have a standard IFC header. The correlation header is added for accounting, performance, audit, and monitor records. The trace header is added for serviceability records.

### 1.12.2 BMC Mainview for DB2 z/OS and DB2 Traces

Users of BMC may easily start a trace from a MAINVIEW AutoOPERATOR console. Submitting the request invokes the trace data collection service (ATRAC).

DB2 traces are activated and terminated transparently to the user as needed to satisfy trace requests. When MVDB2 first establishes a connection to a target DB2 subsystem (BMC Product Address Space PAS startup or at DB2 startup), it uses the Instrumentation Facility Interface (IFI) to reserve one OPx destination (where OPx is an internally defined OP1 to OP8 DB2 buffer). The PAS has a DB2 thread but it is inactive.

Multiple traces can be started with different selection criteria that are specified with the ATRAC request. A maximum of four detail traces per DB2 subsystem and multiple summary traces (all summary traces share a single OPx destination) can be requested. If four detail traces and 20 summary traces are started for the same DB2 subsystem, five (maximum) OPx destinations are used. DB2 has a maximum allowable number of eight OPx destinations for the entire DB2 subsystem. The MVDB2 trace maximum limits the number of trace destinations that are used for the target DB2 by the PAS.

For each detail trace started, a separate DB2 trace is started by the following internal command:

```
-STA TRACE (MON) IFCID (x, x, x . . . ) DEST (OPx)
```

The type of detail trace that is specified with the ATRAC request (TYPE=D | SQL | SCAN | IO | LOCK | DDF | DDFVTAM) determines the set of IFCIDs that is collected:

- D (detail) collects the most important IFCIDs, including all exception events.
- SQL includes all detail events plus SQL statements.
- SCAN includes all detail events plus scan and SQL events.
- IO includes all detail events plus I/O information.
- LOCK includes all detail events plus lock requests.
- DDF includes all detail events plus all DDF events( except those related to VTAM.)
- DDFVTAM includes all detail events plus VTAM-related DDF events.

#### 1.12.2.1 IFCID's and overhead considerations for trace levels

Trace type	Data collected
<b>SUMMARY</b> IFCID's: 001 002 003 090 172 196 225 239	accounting level data This trace type has minimal overhead, because DB2 accounting traces are usually active.
<b>DETAIL</b> IFCID's: 020 023 024 025 029 030 031 044 045 051 052 056 057 068 069 070 071 072 073 074 084 085 088 089 095 096 107 108 109 110 111 112 125 140	adds major events This trace type has relatively low overhead because of selective performance tracing. DETAIL includes the same accounting data as SUMMARY.

Trace type	Data collected
167 168 169 170 171 172 173 177 190 196 213 214 215 216 218 224 226 227 237 337	
<b>SQL</b> IFCID's: 022 053 055 058 059 060 061 062 063 064 065 066 233 247 269 270 272 273 305 311 324 325 350 351 352	adds SQL statements to DETAIL  This trace type is the most useful level for application tuning. This group is included with DETAIL by default.
<b>SCAN</b> IFCID's: 013 014 015 016 017 018 026 027 028 221 222 231	adds scans to DETAIL  This trace level has more overhead because of the potentially large number of scans, but it gives information on index usage and data access paths otherwise not available.
<b>IO</b> IFCID's: 006 007 008 009 010 127 128	adds I/O to DETAIL  This trace level has more overhead because of the additional events traced (more storage is used to hold the large number of events). However, this information is valuable to show the status of the table organization by evaluating GETPAGE to READ I/O ratios.
<b>LOCK</b> IFCID's: 021 211 212 223 251 259	adds lock requests to DETAIL  This trace level has potentially the greatest amount of overhead because of the large number of additional events traced (more storage is used to hold these events). It should be used only if locking is a problem.  Note: The GROUPSQL option can be used to combine identical SQL statements processed in sequence to reduce buffer storage requirements for long-running batch jobs. For example, this option could combine many FETCH statements into one line with summary counts.
<b>DDF</b> IFCID's: [157] 158 159 160 161 162 163 183	adds all DDF events except VTAM-related DDF events to DETAIL  This trace level has more overhead because of additional events traced.
<b>DDFVTAM</b> IFCID's: 164 165 166	adds VTAM-related DDF events to DETAIL  This trace level has more overhead because of additional events traced.

### 1.12.3 BMC Performance Reporter

The SMF data is processed by batch job DPRSMF, which is a built-in feature of BMC Mainview for DB2. The purpose of DPRSMF is to extract DB2 SMF statistics, accounting, and audit records and load this data into the performance data tables in Performance Reporter. The DPRSMF job stream can be tailored to produce accounting and statistics reports during the daily SMF extract, either in addition to, or in place of, loading the data into DB2 tables. The extract files can also be saved for later reporting with the DPRSMFR job.

DPRSMFR maintains following set of DB2 tables:

Performance Reporter Table	Description

<b>DMRSTAT</b> — statistics table	One statistics record is created from each pair of SMF 100 records. This record is further processed to create a delta record, showing the changes in values during this statistics interval. Each row in the DMRSTAT table represents information about one statistics interval within DB2.
<b>DMRSTDF</b> —DDF statistics table	DMRSTDF is an optional table for distributed statistics. Each row in the table represents information about one DDF location for one statistics interval
<b>DMRSBFD</b> —pool detail buffer statistics table	DMRSBFD is an optional table for detailed statistics per buffer pool. One statistics record is created from each pair of SMF 100 records. This record is further processed to create a delta record, showing the changes in values during this statistics interval. Each row in the table represents buffer statistics for one pool for one statistics interval within DB2.
<b>DMRACxxx</b> — accounting detail and summary tables	One accounting record is created from each SMF 101 record that is received. Each row in the DMRACDTL table represents one transaction or thread within DB2, or the number of DDF/RRSAF threads represented by one rollup accounting record.  Each row in a DMRACSUM or DMRACSM2 table represents the aggregate number of the threads summarized according to the defined summary keys for the defined summary interval. For the DMRACSUM or DMRACSM2 tables, nonsummary key IDs are set to the last value processed, counts are totals, and any snapshot values are set to maximums.
<b>DMRABxxx</b> — accounting buffer detail and summary tables	Buffer accounting detail records are created from SMF 101 records that were created with accounting class 7/8 active.  One accounting record is created from each SMF 101 record received. Each row in the DMRABxxx table represents one transaction or thread within DB2.
<b>DMRADxxx</b> —DDF accounting detail and summary tables	DMRADxxx is an optional table for distributed processing accounting records. Each row in the table represents information about one DDF location for one accounting record.
<b>DMRAPxxx</b> — package accounting detail and summary tables	Package accounting detail records are created from SMF 101 records that were created with accounting class 7/8 active.
<b>DMRAUSUM</b> —audit summary table	The DMRAUSUM table is used to produce the Audit Summary and Audit Detail Reports. It requires audit classes 1 through 8 and counts occurrences of IFCIDs 23, 24, 25, 55, 83, 87, 140-145, and 169. It also contains category and type information.
<b>DMRAUFAL</b> — authorization failures table	The DMRAUFAL table is used to produce the Authorization Failures Report. It requires audit class 1 and defines all the fields in IFCID 140.
<b>DMRAUGRV</b> — authorization control table	The DMRAUGRV table is used to produce the Authorization Control - GRANTS/REVOKEs Report. It requires audit class 2 and defines all the fields in IFCID 141.
<b>DMRAUDDL</b> —DDL access table	The DMRAUDDL table is used to produce the DDL Access Report. It requires audit class 3 and defines all the fields in IFCID 142.
<b>DMRAUDML</b> —DML access table	The DMRAUDML table is used to produce the DML Access Report. It requires audit classes 4 and 5 and defines all the fields in IFCIDs 143 and 144.
<b>DMRAUDMB</b> —DML at BIND table	The DMRAUDMB table is used to produce the DML at BIND Report. It requires audit class 6 and defines all the fields in IFCID 145.
<b>DMRAUCHG</b> — authorization ID change table	The DMRAUCHG table is used to produce the Authorization ID Change Report. It requires audit class 7 and defines all the fields in IFCIDs 55, 83, 87, and 169.

## 1.12.4 Compuware Strobe for DB2

The Strobe SQL Agent now utilizes DB2's Instrumentation Facility (IFI) to capture data about any SQL executing during a measurement. This has enabled Strobe to move most of the processing from the user's address space to the Strobe SQL Agent Address Space. The enhanced architecture provides Strobe with a method to capture comprehensive SQL Statistical information, Stored Procedure activity, and Trigger activity. This information is available on iStrobe Reports.

Strobe allows you to enter filtering constraints for IFI if you want to limit the amount of DB2 trace data Strobe collects for SQL.

### 1.12.4.1 Specifying Data Collector Options

You rarely need to use the following command operands. Under normal circumstances, Strobe invokes Strobe options in a configuration appropriate for the environment.

You must specify a data collector option if your installation uses a nonstandard name for any of the subsystems that Strobe supports, unless your Strobe system programmer has provided an alias for the Strobe data collector during system installation.

DB2

DB2|NODB2 is the keyword for Strobe for DB2. You may follow it with these keyword subparameters:

- SQLATTR|NOSQLATTR determines whether Strobe activates the attribution of DB2 SQL CPU activity.
- CAPTURE|NOCAPTURE controls whether Strobe collects DB2 SQL CPU usage.
- DB2CLIST=('filterIDvalue'|'filterIDvalue','filterIDvalue',...) limits the amount of trace information collected using the IBM DB2 instrumentation facility interface (IFI). You should be familiar with IFI and its command requirements before using this feature. You must surround values containing lowercase characters with two pairs of single quotes in order to retain the lowercase. For example, entering "DB2-value" will retain the lowercase of "value".

The filterID portion of 'filterIDvalue' is a prefix ID where:

-P = Plan

-A = AuthIDs

-C = Corrids

-L = Locations

-U = Enduser IDs

-T = Enduser Transactions

-W = Enduser Workstations

-\* = Freeform text string.

For example:

```
DB2CLIST=('Aauthid1','Ccorrid1','Lloc1,loc2,loc3','Userid', 'Ttran1,tran2','WWST4711')
```

For information on these options, see the Strobe for DB2 option manual.

To suppress the collection of DB2 data while measuring within a DB2 region, enter the following:

```
ADD MYJOB, PROGRAM=MYPROG, NODB2
```

DDFRQSTR

DDFRQSTR is the keyword that determines how SQL data is collected for the DDF SQL Statement by DBRM by Requester report. The requester is the source of the SQL that is executing in the DDF address space. You can assign either one or two of the following values:

- REQLOC—Produces the DDF SQL Statement by DBRM by Requester report by location, which is either the IP address or LU name of the requester.
- CORRID—Produces the DDF SQL Statement by DBRM by Requester report by correlation ID.
- AUTHID—Produces the DDF SQL Statement by DBRM by Requester report by authorization ID.
- EUUSER—Produces the DDF SQL Statement by DBRM by End User USERID report.
- EUWORK—Produces the DDF SQL Statement by DBRM by End User Workstation report.
- EUTRAN—Produces the DDF SQL Statement by DBRM by End User Transaction report.

#### Examples

Use this syntax to assign one value:

```
DDFRQSTR={REQLOC | CORRID | AUTHID | EUUSER | EUWORK | EUTRAN}
```

Use this syntax to assign two values, where value 1 and value 2 each represent one of the values specified in the previous syntax. Notice that value 1 and value 2 are separated by a comma and are enclosed by a single quote.

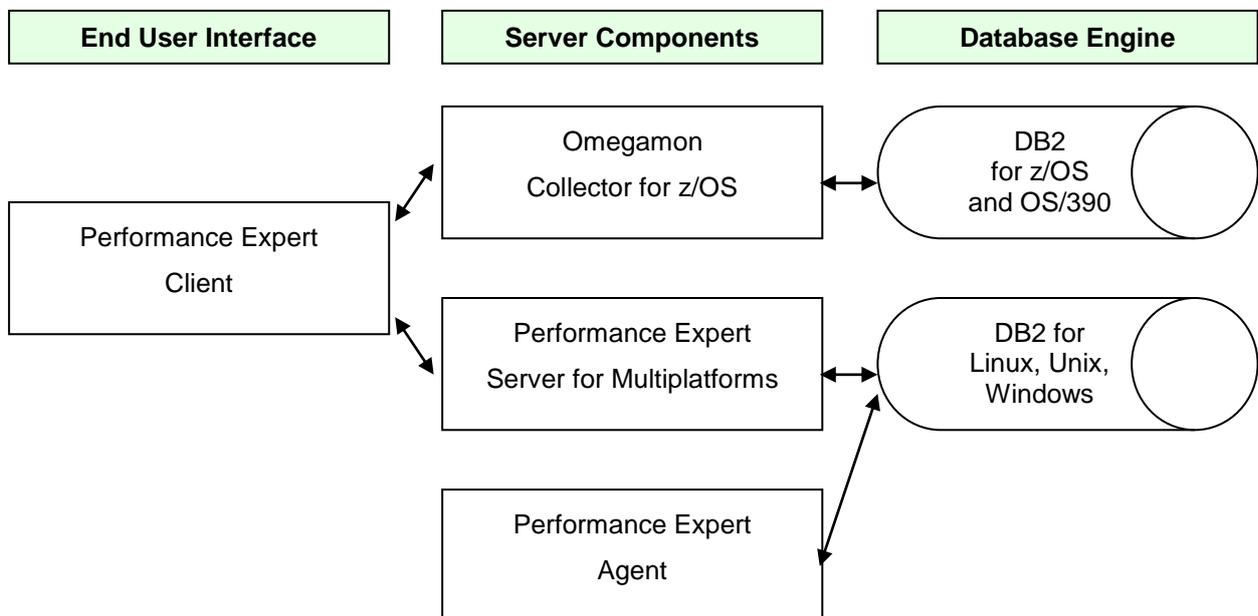
```
DDFRQSTR={'value 1, value 2'}
```

Note: This parameter is set at installation time. The default value is DDFRQSTR=REQLOC. You can temporarily override the value of this parameter when submitting a measurement request, by specifying a different value for DDFRQSTR in the OTHER PARAMETERS field of the Strobe - Add Active Request panel.

For more information about this parameter, see the Strobe for DB2 option manual.

### 1.12.5 OMEGAMON XE for DB2 PE

OMEGAMON XE for DB2 PE consists of several interacting components. The components vary dependent on the platform in use.



The main components of OMEGAMON XE for DB2 PE are:

- Performance Expert Client
- OMEGAMON Collector for z/OS
- Performance Expert Server for Multiplatforms
- Performance Expert Agent for DB2 Connect Monitoring (PE Agent)

Before you can produce an OMEGAMON report or monitor function, you must collect data from DB2 as input. The DB2 Instrumentation Facility Component (IFC) provides a trace facility that is used to record DB2 data and events.

Use any of the following methods to collect data for OMEGAMON reports and traces:

- The Workstation Online Monitor to manage DB2 traces.
- The ISPF Online Monitor to configure a Collect Report Data (CRD) task that collects report data.
- The Data Warehouse to create a process to collect report data.
- DB2 installation system parameters to start traces at DB2 startup time.
- DB2 -START TRACE commands from the console.
- The Collect Report Data Batch program, which is a flexible and resource-efficient way to collect and post-process report data.
- The Near-Term History Data Collector, which can store collected data in sequential data sets.

### 1.12.6 SAP and DB2 for z/OS

SAP is not a DB2 monitor or a tool – that’s for sure. But it provide a bucket of tools worthy to be mentioned.

SAP system are well featured with performance and monitoring tools. Nevertheless SAP lacks some of the features available in stand-alone DB2 monitors. For example, reporting the subsystem-wide statistics data for any given period (DB2 PE Statistics report set) or post-processing of various DB2 performance traces (DB2 PE Record Trace report set) should be obtained by other means.

The performance data that is most relevant to SAP systems is provided by the SAP integrated Database Performance Monitor, which is transaction DBACOCKPIT. Administrators may control activation of DB2-internal trace using the DBACOCKPIT “transaction pool” for administrative transactions. The SAP Collector Setting TON and TOFF within the Configuration Section controls activation or deactivation of DB2 traces.

SAP’s monitor uses the stored procedure SAPCL to gather DB2 performance data from the Instrumentation Facility Interface (IFI). In previous releases, rfcoscol provided this function. SAPCL is accompanied by an alert router process that is controlled by the stored procedure. SAPCL can also be used to monitor remote DB2 subsystems.

With regard to performance monitoring, SAPCL serves the following purposes:

- Provides current snapshot information of the overall system
- Allows you to submit DB2 commands
- The alert router process catches database alerts, which are raised by DB2, and propagates them to the SAP system

Once activated the SAP system tries to ensure that the relevant DB2 traces are always on. These traces are source of performance indicators, which allow you to tune the DB2 subsystem directly and to get appropriate diagnostics information in case of performance problems. A periodic SAP job (called RSDB2\_RUN\_HOURLY\_REMOTE) runs at the startup of the SAP system and once every hour. Its goal is to start both accounting trace classes 1, 2 and 3, IFCID 318 of performance trace class 30 and IFCID 199 of statistics trace class 8. On request IFCID 255, which is contained in Statistics Trace class 6, provides information on DBM1 virtual storage statistics. The DB2 Message Log displays the DB2 console messages provided by IFCID 197

The SAP-internal monitor program SAPCL connects to the DB2 subsystem for which the performance data should be gathered. As soon as a request for performance data comes in, SAPCL calls DB2 via the IFI interface. The collected data is examined and written into regular or temporary tables. Then the calling transaction reads these data from the corresponding tables. To keep the regular SAP monitor tables from growing excessively, the content of the tables is automatically deleted. Regular tables mainly hold database alerts. SAPCL inserts performance indicators into temporary tables to satisfy synchronous requests for performance data.

SAP systems with DB2 z/OS Database Servers provide following diagnostic trace facilities:

- DB2 Connect CLI Trace
- IFI DB Trace (SQL TRACE, LOCK TRACE, DATA TRACE)
- DBSL SQL Trace
- IFI Data Collector Trace (Tracing Data for SAPCL)
- z/OS System Log/z/OS System Log (SM21)
- System Trace (ST01)
- ABAP Runtime Analysis (SE30)
- SQL Trace (ST05)

### 1.13 List of available IFCID's

001 - System Statistics	045 - Lock Resume	086 - Signon Start
002 - DB2 Statistics	046 - IBM Service Record	087 - Signon End
003 - Accounting	047 - IBM Service Record	088 - Synch Start
004 - Trace Start	048 - IBM Service Record	089 - Synch End
005 - Trace Stop	049 - IBM Service Record	090 - DB2 Command Start
006 - Read I/O Start	050 - IBM Service Record	091 - Command End
007 - Read I/O Stop	051 - IBM Service Record	092 - AMS Command Start
008 - Write I/O Synch	052 - IBM Service Record	093 - IBM Service Record
009 - Write I/O	053 - SQL	094 - IBM Service Record
010 - Write I/O Asynch	Describe/Commit/	095 - Sort Start
011 - Validate Exit	Rollback/Remote Stmt	096 - Sort End
012 - Edit Exit to Encode	055 - Set SQLID	097 - AMS Command End
013 - Hash Scan Input Start	056 - IBM Service Record	098 - IBM Service Record
014 - Hash Scan End	057 - IBM Service Record	099 - IBM Service Record
015 - Index Scan Begin	058 - End SQL	100 - IBM Service Record
016 - Insert Scan Begin	059 - Fetch Start	101 - IBM Service Record
017 - Sequential Scan Begin	060 - Select Start	102 - IBM Service Record
018 - Scan End	061 - Insert/Update/Delete	103 - SOS Off
019 - Edit Exit to Decode	Start	104 - Log Data Set
020 - Lock Summary	062 - DDL Start	105 - DBID/OBID
021 - Lock Detail	063 - SQL Statement	Translation
022 - Minibind	064 - Prepare Start	106 - System Parameters
023 - Utility Start	065 - Open Cursor	107 - Open/Close
024 - Utility Change	066 - Close Cursor	108 - Bind Start
025 - Utility End	067 - Accounting	109 - Bind End
026 - IBM Service Record	068 - Rollback Start	110 - Bind Free Start
027 - Sort Workfile Records	069 - IBM Service Record	111 - Bind Free End
028 - Sort Phase Detail	070 - Commit Phase 2 Start	112 - Thread Allocate
029 - EDM Request Start	071 - IBM Service Record	113 - Agent Allocate
030 - EDM Request End	072 - Create Thread Start	114 - Archive Wait Start
031 - EDM Full	073 - Create Thread End	115 - Archive Wait End
032 - Log Wait Start	074 - Terminate Thread	DASD
033 - IBM Service Record	Start	116 - Archive Wait End
034 - Log Read Start	075 - Terminate Thread End	Tape
035 - Log Read End	076 - End of Memory Start	117 - Archive Read Start
036 - Log Non I/O Start	077 - End of Memory End	118 - Archive Read End
037 - Log Non I/O End	078 - End of Task Start	119 - BSDS Write Start
038 - Active Write Start	079 - End of Task End	120 - BSDS Write End
039 - Active Write End	080 - IBM Service Record	121 - IBM Service Record
040 - Archive Write Start	081 - IBM Service Record	122 - IBM Service Record
041 - Archive Write End	082 - Identify Start	123 - SRV Record
042 - Checkpoint Start	083 - Identify End	124 - SQL Statement
043 - Checkpoint End	084 - Prepare Start	Record
044 - Lock Suspend	085 - Prepare End	125 - RID Pool Processing

126 - Log Buffer Write	184 - DC Communication Buffers	225 - System Storage Usage
127 - Page Wait I/O In Prog (Start)	185 - READs Data Capture Start	226 - Page Latch Contention Start
128 - Page Wait I/O In Prog (End)	186 - IBM Service Record	227 - Page Latch Contention End
129 - CI-S Obtained via IFI Reads	188 - READs Data Capture End	228 - Archive Deallocation Start
140 - Audit Auth Failures	190 - IBM Service Record	229 - Archive Deallocation End
141 - Audit DDL Grant/Revoke	191 - DDM Level 6B Objects	230 - Group Buffer Pool Attributes
142 - Audit DDL Create/Alter/Drop	192 - DDM Level 6A Header Errors	231 - Parallel Group Task Time
143 - Audit First Write	193 - UOW/SQLCODE Mismatch	233 - Call User Routine
144 - Audit First Read	194 - Invalid SNA FMH-5 Received	234 - Calling Agent Auth IDs
145 - Audit DML Statement	195 - SQLDA Discrepancy	236 - DDF SNA XLN Protocol Error
146 - User Record	196 - Timeout Data	237 - Set Current Degree
147 - Thread Summary	198 - Buffer Manager Page Access	238 - IBM Service Record
149 - Resource Locking	199 - Buffer Pool Statistics at Data Set Level	239 - Overflow Package/DBRM
150 - Thread Locking	201 - Alter Buffer Pool	247 - SQLDA Data and Input Host Variable Data
151 - User Record	203 - DDF Heuristic COMMIT/ROLLBK	248 - IBM Service Record
152 - User Record	204 - DDF Partner Cold Start	249 - EDM Pool Invalidate DBD
153 - User Record	205 - DDF Warm Start Log Name Error Information	250 - Connect/Rebuild Connect/Disconnect Group Bpool
154 - User Record	206 - DDF Protocol Error	251 - Buffer Manager PSET/Part P-Lock Request
155 - User Record	207 - DDF Heuristic Damage	252 - IBM Service Record
156 - User Record	208 - DDF Syncpoint Protocol Error	254 - Coupling Facility Cache Structure Statistics
[157] - DRDS RDS Interface, removed from DB2 V10	209 - DDF Syncpoint Comm Failure	255 - Buffer Refresh Due to XI
158 - DRDS CNV Interface	210 - Warm Start Log Name Change	256 - Alter Group Buffer Pool
159 - DRDS Req Site Data	211 - Claim Data	257 - IRLM Notify Req Detail
160 - DC Requester	212 - Drain Data	258 - Data Set Extend Activity
161 - DC Server	213 - Drain Lock Wait Start	259 - Buffer Manager Pg P-Lock Req
162 - DTM Request	214 - Drain Lock Wait End	260 - IBM Service Record
163 - DTM Respond	215 - Claim Count 0 Wait Start	261 - Group Buffer Pool Checkpoint
164 - IBM Service Record	216 - Claim Count 0 Wait End	262 - GBPOOLT Castout Threshold Processing
165 - IBM Service Record	217 - Storage Pools	263 - Page Set and Partition Castout Detail
166 - IBM Service Record	218 - Lock Avoidance Summary	265 - IBM Service Record
167 - Conv Alloc Req Queued	219 - Utility LISTDEF List Information	266 - IBM Service Record
168 - IBM Service Record	220 - Utility Data Set Information	267 - CF Rebuild/Alter/Start
169 - DIST Authid Translation	221 - Parallel Group Execution	268 - CF Rebuild/Alter End
170 - Suspend of Agent	222 - Parallel Group Elapsed Time	269 - Trusted/Context Trace
171 - IBM Service Record	223 - Lock Avoidance Detail	
172 - Deadlock Data	224 - Select Procedure Bypassed	
173 - Class 2 Time		
174 - Arch Log CMD Sus Start		
175 - Arch Log CMD Sus End		
177 - Package Allocation		
178 - IBM Service Record		
179 - IBM Service Record		
180 - DC Communication Buffers		
181 - IBM Service Record		
182 - IBM Service Record		
183 - DRDS RDS/SCC Interface		

- |   |  |  |
|---|--|--|
| 270 – Row/Col Perm's (V10)                      | 337 - Lock Escal. Occurrences (DB2 V8)             | 371 - Database Close Information                                 |
| 271 - Row Level and Column Level Access Control | 342 - WF/TEMP DB Usage                             | 373 - DSNHDECP location and name at startup                      |
| 272 - Associate Locators                        | 343 - MAXTEMPS Limit/Exceeded                      | 377 - Pseudo Delete Daemon Cleanup                               |
| 273 - Allocate Cursor                           | 345 - Trace Data / SP/UDF                          | 378 - Accel. Call Event Begin (DB2 10 or later)                  |
| 305 - Table Check Constraint                    | 346 - Package/DBRM Detail                          | 379 - Accel. Call Event End (DB2 10 or later)                    |
| 311 - Global Temp Table Usage                   | 350 - SQL Statement                                | 380 - Stored Procedure Detail Record                             |
| 313 - Uncommitted Unit of Recovery              | 351 - Wait TCPIP LOB                               | 381 - UDF Detail Record  |
| 314 - Authorization Exit Parameters             | 353 - IBM Service Record                           | 382 - Records suspend operations for parallel task               |
| 316 - SQL Statement Statistics                  | 354 - IBM Service Record                           | 383 - Records resume operations for parallel task                |
| 317 - SQL Statement String                      | 357 - Beginning of an Index I/O Parallel INSERT    | 385, 386, 397, 398 and 399 - IBM service                         |
| 318 - Dynamic Statement Cache Statistics Switch | 358 - End of an Index I/O Parallel INSERT          | 401 - Static Statements in EDM Pool                              |
| 319 - Audit Security Record                     | 359 - Index Page Split                             | 402 - System Profile - Monitoring Statistics (DB2 V10 and later) |
| 321 - Force-at-Commit Begin                     | 360 - Incremental query binds (V10)                | 497 - Non Nested Statement ID Record                             |
| 322 - Force-at-Commit End                       | 361 - Audit Admin Authorities                      | 498 - UDF Statement ID Record                                    |
| 324 - Function Resolution                       | 362 - Start Trace and Stop Trace with Audit Policy | 499 - Stored Procedure Statement ID Record                       |
| 325 - Trigger Activation                        | 363 - Parallel Straw Model Performance Trace       |  |
| 329 - IXL Suspensions                           | 364 - Address Space Creation and Termination       |  |
| 330 - Active Log Space Shortage                 | 365 - DRDA Remote Location Statistics              |  |
| 331 - IBM Service Record                        | 369 - Aggregated Accounting Statistics             |  |
| 332 - IBM Service Record                        | 370 - Database Open Information                    |  |
| 333 - IBM Service Record                        |  |  |
| 335 - System Event Stalled                      |  |  |

IFCID 157 is removed from DB2 for z/OS Version 10.

## 1.14 IFCID mappings

Each trace class captures information on several subsystem events. These events are identified by many instrumentation facility component identifiers (IFCID's). The IFCID's are described by the comments in their mapping macros, contained in prefix.SDSNIVPD. A description of all IFCID's is included in prefix.SDSNIVPD(DSNWMSGs), which is also shipped with DB2. Users may load this information to user-tables for easy access on IFCID description.

In early days DSNWMSGs was placed in SDSNSAMP but nowadays the LRECL 80 of DSNSAMP are just too short, so IBM separated this member to DSNIVPD. The corresponding mapping macros are in SDSNMACS. Find below a table of IFCID's and macros.

IFCID	Mapping Macro
0001	system statistics records and is mapped by macro DSNDQWST subtype=0, and DSNDQWS0
0002	system statistics records and is mapped by macro DSNDQWST subtype=1, and DSNDQWS1.
0003	accounting records and is mapped by macro DSNDQWAS and DSNDQWA0.
0004 - 0057	mapped by DSNDQW00

0058 - 0139 (except 106)	mapped by DSNDQW01
0106	mapped by DSNDQWPZ
0140 - 0200	mapped by DSNDQW02
0201 - 0249 (except 202, 230)	mapped by DSNDQW03
0202	mapped by DSNDQWS2
0230	mapped by DSNDQWS3
0239	mapped by DSNDQWAS subtype=1
0250 – 360	mapped by DSNDQW04
361 and up	Mapped by DSNDQW05

## 1.15 Changes from DB2 for z/OS Version 10

### 1.15.1 SMF subtype field defined as a two-byte value

The SMF subtype field should be defined as a two-byte value. Previously, DB2 mapped the field SM10xSTF as a single-byte field that corresponds to the high half of the standard two-byte field location. In Version 10, SM10xSTF is changed to a two-byte field. This change results in the removal of unused field SM10xRI from the mapping. This change might require change to applications that parse SMF.

### 1.15.2 Changed default OP buffer size

The online performance (OP) buffer size in Version 10 defaults to 1 MB unless otherwise specified.

### 1.15.3 New IFCID's

IFCID	Trace	Class	Mapping macro	Description
<b>64-bit support</b>				
<b>0360</b>	Performance	3, 10	DSNDQW04	Records information about queries that are incrementally bound because parallelism was chosen in packages that were created before DB2 Version 10.
<b>Row and column access control</b>				
<b>0270</b>	Audit	11	DSNDQW04	Records information about a row permission or a column mask when it is created, dropped, or altered.
<b>Index I/O parallelism</b>				
<b>0357</b>	Performance	4	DSNDQW04	Records the start of an instance of the use of index I/O parallelism for indexes on a table.
<b>0358</b>	Performance	4	DSNDQW04	Records the end of an instance of the use of index I/O parallelism for indexes on a table.
<b>New DB2 authorities</b>				
<b>0361</b>	Audit	11	DSNDQW05	Records information for auditing administrative authorities.
<b>0362</b>	Audit	11	DSNDQW05	Records information about the start of an audit trace with AUDITPOLICY.
<b>Parallelism enhancements</b>				
<b>0363</b>	Performance	8	DSNDQW05	Records information about a new workload distribution method for parallel group execution.
<b>Reduce need for frequent explicit REORGs</b>				
<b>0359</b>	Performance	4	DSNDQW04	Records an index page split.
<b>Enhanced monitoring support</b>				

IFCID	Trace	Class	Mapping macro	Description
0400	Monitor	29	DSNDQW05	A switch that enables collection of detailed information about static SQL statements that are in the EDM pool.
0401	Monitor	29	DSNDQW05	Records detailed information about static SQL statements that are being tracked in the EDM pool. Important: If you are migrating to DB2 Version 10, to retrieve IFCID 0401 information for packages, you need to bind those packages in Version 10 new-function mode.
0402	Monitor	29	DSNDQW05	Records information about a profile warning or exception condition that is related to system monitoring for threads and connections.
<b>Other IFC enhancements</b>				
0364	Global	3	DSNDQW05	Records address space creation and termination.
0365	Statistics	7	DSNDQW05	Records detailed statistics about the remote locations with which a DB2 subsystem communicates using the DRDA® protocol.
0373			DSNDQW05	Records the location and name of the dsnhdecp (application defaults) module that was loaded by the attached subsystem at DB2 startup. This record is available only through an IFI READS call.

#### 1.15.4 Changed IFCID's

IFCID	Description of changes
<b>64-bit support</b>	
0015, 0016, 0017, 0018, 0277, 0278	Four-byte fields that contain pointers to storage that is now above the 2 GB bar are no longer used. They are replaced by eight-byte fields.
0027, 0038, 0096	Four-byte fields that contain record counters are no longer used. They are replaced by eight-byte fields.
0225	Fields are added to record more storage statistics.
<b>Access currently committed data</b>	
0002	Fields are added to record the number of rows that were skipped or accessed by read transactions when currently committed read behavior was in effect for fetch operations.
<b>Backup and recovery utility enhancements</b>	
0024	A field is changed to record the SEQCOPY phase of the COPY utility.
<b>Buffer pool enhancements</b>	
0201, 0202	Fields values are added to indicate when the PGSTEAL attribute for a buffer pool is NONE.
<b>Dynamic statement cache literal replacement</b>	
0002, 0361	Fields are added to indicate when cached statements are reused or not reused due to literal replacement.
<b>Eliminate private protocol</b>	
157	This IFCID is removed.
<b>Enhanced monitoring support</b>	
0058	Fields are added to record the statement type, statement execution identifier and statement level performance metrics.
0063, 0350	Fields are added to return the statement type, statement execution identifier and original source CCSID of the SQL statement.
0065	A field is added to indicate whether implicit commit is requested when a cursor is closed.
0066	A field is added to indicate whether a close operation is an explicit or implicit.

IFCID	Description of changes
0124	Fields are added to return the statement execution identifier and statement type.
0172, 0196, 0337	Fields are added to return the statement execution identifier and statement type when a deadlock or timeout condition is detected.
0316	<ul style="list-style-type: none"> <li>When a statement is removed from the dynamic statement cache, an IFCID 0316 record will now be written that describes the statement and statistics about the statement.</li> <li>Statistics fields are increased from four bytes to eight bytes.</li> </ul>
<b>New DB2 authorities</b>	
0140, 0141	Field values are added for the new authorities.
<b>Row and column access control</b>	
0062	Constants are added for new SQL statements and object types.
0140	Constants are added for privileges that are checked for row or column access control.
0145, 0350	A field is added to indicate whether the target table of the SQL statement uses row or column access control.
<b>Temporal tables</b>	
0021, 0044, 0150, 0172	Constants are added for the new index key lock.
<b>Miscellaneous changes</b>	
<b>Trace records written to SMF</b>	If subsystem parameter SMFCOMP is set to ON, everything after the SMF header is compressed. A bit in the SMF header indicates whether a record is compressed.
0001	<ul style="list-style-type: none"> <li>Descriptions of QDST fields are updated and clarified.</li> <li>New fields are added to record information about the number of database access threads that are active because they are in packages that are bound with RELEASE(DEALLOCATE).</li> </ul>
0001, 0002, 0202, 0217, 0225	When traces for these IFCIDs are enabled, the trace records are now written at fixed, one-minute intervals.
0002	<ul style="list-style-type: none"> <li>Pairs of fields that record storage use in whole megabytes and fractions of megabytes are replaced with single fields that record storage use in kilobytes.</li> <li>Additional fields to track EDM pool storage usage are added.</li> <li>Several counters in macros DSNDQBGL and DSNDQBST are increased from four bytes to eight bytes.</li> <li>Field QISESPLR has been removed.</li> <li>IFCID 0002 records can now contain up to 25 QBST repeating groups and 25 QBGL repeating groups. If additional QBST data exists, it is written in an additional IFCID 0002 record that contains only QBST repeating groups. If additional QBGL data exists, it is written in an additional IFCID 0002 record that contains only QBGL repeating groups. Therefore, the maximum number of IFCID 0002 records for a statistics interval is three.</li> <li>Distributed data facility statistics records, which are mapped by DSNDQLST, are changed as follows: <ul style="list-style-type: none"> <li>Two-byte numeric fields are extended to four bytes.</li> <li>Fields are reordered for better clarity.</li> <li>Fields that refer exclusively to DB2 private protocol are removed.</li> <li>Several pairs of fields contained similar information. For those fields, one field now contains the information from both fields, and the other field has been removed.</li> </ul> </li> </ul>
0002, 0003	<ul style="list-style-type: none"> <li>The number of DESCRIBE statements in field QXDESC now matches the number of DESCRIBE statements in the user application.</li> <li>Several counters in macro DSNDQXST are increased from four bytes to eight bytes.</li> </ul>
0002, 0003, 0148, 0316, 0401	Fields that track the use of RID lists for index access are added or changed to track RID list overflow to a work file.

IFCID	Description of changes
0003	<ul style="list-style-type: none"> <li>• Fields are added to describe: <ul style="list-style-type: none"> <li>• The number of parallel query child agents in a trace roll-up record</li> <li>• Whether a trace roll-up record contains parallel query roll-up data</li> <li>• Information about waits for locks</li> </ul> </li> </ul>
0003, 0147, 0148	<ul style="list-style-type: none"> <li>• The following types of data are eligible for DDF and RRSAF rollup: <ul style="list-style-type: none"> <li>• Distributed accounting data</li> <li>• MVS™ and DDF (QMDA) accounting information</li> <li>• IFI accounting data</li> </ul> </li> </ul>
0003, 0148	<ul style="list-style-type: none"> <li>• A data section mapped by DSNDQWAR is added for rollup accounting correlation.</li> <li>• Distributed data facility accounting records, which are mapped by DSNDQLAC, are changed as follows: <ul style="list-style-type: none"> <li>• Two-byte numeric fields are extended to four bytes.</li> <li>• Fields are reordered for better clarity.</li> <li>• Fields that refer exclusively to DB2 private protocol are removed.</li> <li>• Several pairs of fields contained similar information. For those fields, one field now contains the information from both fields, and the other field has been removed.</li> </ul> </li> </ul>
0021	The field that records lock types is changed to include the utility object lock, whose lock type is X'37'.
0058	Fields are added to record: <ul style="list-style-type: none"> <li>• Accumulated wait time for a lock request</li> <li>• Accumulated wait time for a latch request</li> <li>• Accumulated wait time for a page latch</li> <li>• Accumulated wait time for a claim count</li> <li>• Accumulated wait for log writers</li> </ul>
0148	Fields are added to record start and end times for the most recent SQL entry for the current nested activity.
0217, 0225	These records have been reorganized for better consumability.
0239	The following changes have been made: <ul style="list-style-type: none"> <li>• Fields are added to record information about waits for locks.</li> <li>• Package detail data is now eligible for parallel query rollup and DDF and RRSAF rollup.</li> <li>• Field QPACSWITCH has been changed so that a return from a nested package no longer counts as a package switch.</li> </ul>
0267	The field that records the type of operation that is performed on the underlying structure of a data sharing group is changed to include rebuild of the lock structure due to restart delay.
0306	The record header is increased from 60 bytes to 128 bytes. Records with the 128-byte header have the string 'V10 ' in the fifth through twelfth bytes.
0316	Fields are added to record: <ul style="list-style-type: none"> <li>• Referenced table name</li> <li>• Transaction name</li> <li>• End-user ID</li> <li>• Workstation name</li> <li>• Accumulated wait time for a latch request</li> <li>• Accumulated wait time for a page latch</li> <li>• Accumulated wait time for a drain lock</li> <li>• Accumulated wait time for a drain during a wait for claims to be released</li> <li>• Accumulated wait for log writers</li> <li>• Timestamp when a statement was inserted into the EDM pool</li> <li>• Timestamp statement statistics were updated</li> </ul> READS filters WQALEUID, WQALEUTX, and WQALEUWN now apply to IFCID 0316.
0342	Fields that record space use for tables and indexes in 4KB blocks are changed to record space use in kilobytes.

IFCID	Description of changes
<b>0343</b>	Fields that record work file use in megabytes are changed to record work file use in kilobytes.

### 1.15.5 Changed format of the IFCID 0002, 0003, 0148, and 0306 trace records

For IFCID 0002, the format of the distributed data facility area, which is mapped by QLST fields, has changed substantially. For IFCID 0003, the format of the distributed data facility area, which is mapped by QLAC fields, has changed substantially. For IFCID 0148, the format of the distributed accounting records has changed substantially. Any application that reads and interprets location statistics must be modified to account for the Version 10 format of IFCID 0002, 0003, and 0148. For IFCID 0306, the definition of field QW0306AD has changed and the size of the area mapped by field QW0306OF has increased. The first eight bytes of field QW0306AD are now "V10 " to indicate that the data is from DB2 Version 10 and that the Version 10 mapping should be used.

Changes to the following IFCIDs might require changes to your trace applications: 0002 0003 0015 0016 0017 0018 0027 0028 0096 0217 0225 0247 0248 0277 0278 0306

### 1.15.6 Changes in IFCID structure documented with OMEGAMON

Following document describes the updates for OMEGAMON® V5.1 products in "Report Reference Manual", SH12-6921-02: <http://www-01.ibm.com/support/docview.wss?uid=swg27020420>.

Following document describes the updates for OMEGAMON® V5.2 products in "Report Reference Manual", SH12-6991-00: <http://www-01.ibm.com/support/docview.wss?uid=swg27040297>.

## 1.16 Changes from DB2 11 for z/OS

### 1.16.1 New IFCID's

IFCID	Trace	Class	Mapping macro	Description
<b>Index management</b>				
<b>0377</b>	Performance	3, 10	DSNDQW04	Records index pseudo delete daemon cleanup.
<b>0106</b>	Audit	11	DSNDQW04	Records information about a row permission or a column mask when it is created, dropped, or altered.
<b>0027</b>	Performance	4	DSNDQW04	Monitors sparse index usage.
<b>Index management</b>				
<b>0382</b> <b>0383</b>				Records {suspend   resume} operations for parallel task.

### 1.16.2 Changed IFCID's

IFCID	Description of changes
<b>Array support</b>	
<b>0002, 0225</b>	Fields are added to record information about storage use by arrays.
<b>Autonomous transactions</b>	
<b>0003, 0239</b>	Fields are added to record information about autonomous transactions, and on their effect on parallel groups.
<b>Application compatibility</b>	
<b>0366</b>	Field values are added to record incompatibilities between Version 11 and previous DB2 versions.
<b>Castout enhancements</b>	
<b>0230, 0256</b>	Fields are added to record class castout queue threshold values, which are based on the number of pages.
<b>Larger RBA and LRSN support</b>	

IFCID	Description of changes
<b>0001, 0032, 0034, 0043, 0114, 0119, 0143, 0144, 0185, 0188, 0203, 0204, 0206, 0207, 0208, 0209, 0235, 0261, 0306, 0313, 0335</b>	Fields that contain RBAs and LRSNs are expanded from 6 or 8 bytes to 10 bytes. Fields other than QW0204UR are moved to avoid changes in other offsets. Field QW0204UR is expanded but not moved. Therefore, the offsets of all fields in IFCID 0204 that follow it are changed.
<b>Parallelism performance enhancements</b>	
<b>0002, 0003, 0316, 0401</b>	Fields are added to track the effect of changes to the degree of parallelism after parallel system negotiation that occurs because of resource constraints.
<b>Temporal support</b>	
<b>0053, 0058, 0060, 0061, 0065, 0316, 0401</b>	Fields are added to indicate the impacts of the CURRENT TEMPORAL BUSINESS_TIME special register, the CURRENT TEMPORAL SYSTEM_TIME special register, and the SYSIBMADM.GET_ARCHIVE built-in global variable.
<b>Miscellaneous changes</b>	
<b>0002</b>	A field is added to record the amount of storage that is allocated to shareable, static SQL statements.
<b>0002, 0003</b>	Fields are added to record the number of pages that are written to disk through group-buffer-pool write-around protocol.
<b>0003</b>	<ul style="list-style-type: none"> <li>• Fields are added to record the wait time and number of waits for parallel queries to synchronize.</li> <li>• Existing fields that record accumulated wait time for write I/O that is done under another thread and the number of wait trace events that were processed for write I/O under another thread now include time that is consumed by DFSMS Media Manager calls when buffers are written to disk.</li> </ul>
<b>0003, 0148, 0239</b>	Sysplex query parallelism is no longer supported. Fields that are related to Sysplex query parallelism are not populated.
<b>0127, 0128</b>	Fields are added to record the time spent waiting for buffer manager force write I/O.
<b>0148</b>	A field is added to record an additional DDF enclave.
<b>0201</b>	Fields are added to record the virtual pool size, minimum pool size, maximum pool size, and frame size before and after the ALTER BUFFERPOOL command.
<b>0217, 0225</b>	Fields are added to record IRLM private storage limits.
<b>0225</b>	<p>The following changes are made:</p> <ul style="list-style-type: none"> <li>• Fields are added to record the common storage that is used by log manager buffers and control structures.</li> <li>• Data is no longer recorded for the number of log manager write buffer frames in auxiliary storage.</li> <li>• Fields that record shareable storage use for SQL statements are moved and expanded to 8 bytes. The corresponding 4-byte fields are no longer used.</li> </ul>
<b>0239</b>	<ul style="list-style-type: none"> <li>• Package detail information is rolled up and associated with a rollup QPAC record if an accounting class 10 trace or a monitor class 10 trace is enabled.</li> <li>• Fields are added to record the wait time and number of waits for parallel queries to synchronize.</li> </ul>
<b>0313</b>	<p>New fields are added to record:</p> <ul style="list-style-type: none"> <li>• User IDs that are longer than 16 bytes</li> <li>• Transaction names that are longer than 32 bytes</li> <li>• Workstation names that are longer than 18 bytes</li> </ul>

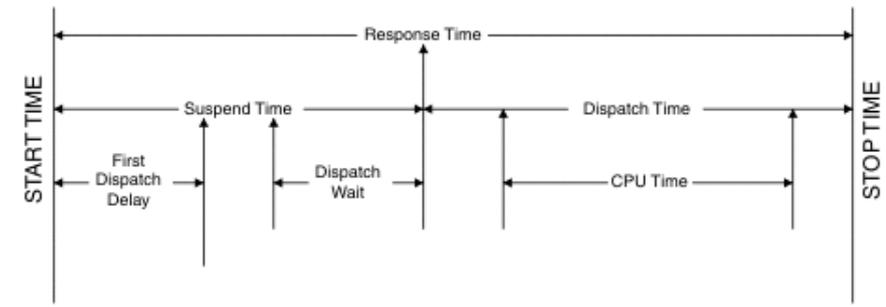
## 1.17 DB2 Monitoring Glossary

Glossary Term or Monitor Field	Description (and Recommendations)
IFI	Instrumentation Facility Interface. A programming interface that enables programs to obtain online trace data about DB2, to submit DB2 commands, and to pass data to DB2.
IFCID	instrumentation facility component identifier. A value that names and identifies a trace record of an event that can be traced. As a parameter on the START TRACE and MODIFY TRACE commands, it specifies that the corresponding event is to be traced.
GTF	Generalized trace facility. A z/OS service program that records significant system events such as I/O interrupts, SVC interrupts, program interrupts, or external interrupts.
Resource Manager	A function that is responsible for managing a particular resource and that guarantees the consistency of all updates made to recoverable resources within a logical unit of work. The resource that is being managed can be physical (for example, disk or main storage) or logical (for example, a particular type of system service).
SMF	System Management Facilities. An optional control program feature that provides the means for gathering and recording information that can be used to evaluate system usage.
Elapsed Time	<p>The diagram illustrates the components of elapsed time. It is structured as a series of nested boxes. The outermost box is labeled 'CICS/IMS Transit Time'. Inside this box is 'Accounting Class 1 Elapsed Time'. Inside that is 'Accounting Class 2 Elapsed Time'. Inside that is 'Class 3 Elapsed Time'. Inside that is 'CPU Time'. To the right of 'CPU Time' are three vertical boxes: 'I/O Suspension', 'Lock/Latch Suspension', and 'other DB2 Time'. To the right of 'other DB2 Time' is a box labeled 'Not-in-DB2-Time (Application)'.</p>
DB2 Unit Switch	<p>To display the amount of time the thread waited to use DB2 services due to the synchronous execution of a unit switch.</p> <p>The following are DB2 services for which unit switch wait elapsed time is accumulated:</p> <ul style="list-style-type: none"> <li>o Open/close data set</li> <li>o SYSLGRNG update</li> <li>o Phase-2 commit for read-only threads originating from TSO or batch</li> <li>o HSM recall data set</li> <li>o Dataspace Manager Services</li> <li>o Define data set</li> <li>o Extend data set</li> </ul>

Glossary Term or Monitor Field	Description (and Recommendations)
	<ul style="list-style-type: none"> <li>o Delete data set</li> </ul> <p>There is no overlap between the elapsed time reported in this field and the other class 3 elapsed times. There also is no overlap between the elapsed time reported in this field and the elapsed time reported for threads with distributed activity.</p> <p>No service waits are associated with phase-2 commit processing under read-only threads originating from CICS or IMS. There is a service wait for any thread performing phase-2 commit processing after an update. Since a log record is forced during the update phase-2 commit processing, however, DB2 captures the log write I/O time and ignores service wait time.</p>
Getpages per I/O Pages read	<p>The ratio of GETPAGE requests to the total pages read from DASD. The total pages read include pages read through synchronous and asynchronous requests.</p>
Other Read I/O	<p>To display statistics for the number of wait trace events processed for waits for read I/O under another thread since this thread began processing in DB2 (for example, a system agent). This field also displays statistics for the accumulated wait time for read I/O that is performed under a thread other than this one. This value is calculated for threads by subtracting the store-clock (STCK) time on entry to event wait from STCK on resume from the event.</p> <p>A large value indicates contention on a busy dataset, control unit or controller. If the problem is due to an I/O bound query, use DEGREE(ANY) to enable parallelism.</p>
Other Write I/O	<p>To show statistics for the number of wait trace events processed for waits for write I/O under another thread since this thread began processing in DB2 (for example, a system agent). This field also displays statistics for the accumulated wait time for write I/O that is performed under a thread other than this one. This value is calculated for threads by subtracting the STCK (store clock) time on entry to event wait from STCK on resume from the event.</p> <p>A large value indicates contention.</p>
Sync. I/O Suspension	<p>= Class 3 suspend time. The elapsed time spent waiting because of synchronous I/O suspensions.</p> <p>Large times indicate that there may have been a change in the access path. The change in the access path and a large number of getpages indicate the need to reorganize the data. (Tablespace Scan using sequential prefetch with prefetch threshold being reached)</p>
Pages/IO	<p>To show the ratio of GETPAGEs to I/O. The algorithm used to calculate the value you see is:</p> $\text{GETPAGE}/(\text{SYNCREAD} + \text{ASYNCREAD})$ <p>where, „GETPAGE“ is the total number of GETPAGEs, „SYNCREAD“ is the total number of synchronous reads, and „ASYNCREAD“ is the total number of asynchronous reads.</p> <p>A low GETPAGES/READ IO value indicates one of the following:</p> <ul style="list-style-type: none"> <li>o An application is seeking single rows. For example, a transaction is performing get unique processing. This is unavoidable, but also desirable.</li> <li>o The buffer pool or pools are too small. An application may actually be able to process many pages per read, but the buffers are too small</li> </ul>

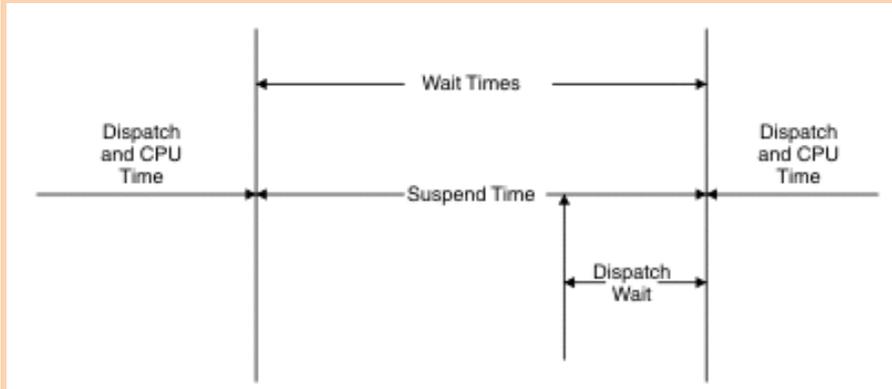
Glossary Term or Monitor Field	Description (and Recommendations)
	<p>and DB2 cannot asynchronously satisfy the requests.</p> <p>Examine applications with GETPAGES/READ IO rates of less than 2.0 to make sure these applications are actually doing random processing. If the applications are not processing randomly, examine buffer pool utilization to determine whether you should increase buffer pool size.</p>
Accounting Reason	<p>The reason accounting was invoked for a thread. Valid values are:</p> <ul style="list-style-type: none"> <li>• APPLICATION ABND (abnormal program termination)</li> <li>• AUTH ID CHANGE (reusable thread with an authorization ID change or one that went through sign-on processing with the same authorization ID)</li> <li>• CANCEL/STOP FRCE (thread was terminated due to DB2 STOP FORCE/CANCEL was issued)</li> <li>• CREATE FAILED (thread failed during create thread processing)</li> <li>• DBAT DEACTIVATE (a database access thread is terminating)</li> <li>• END OF MEMORY (abnormal program termination due to end of memory)</li> <li>• INCOMPLETE (thread was terminated, but a TERMINATE END IFCID for the thread was never encountered by TMON for DB2)</li> <li>• INDOUBT - ABEND (abnormal program termination due to end memory while in an indoubt state)</li> <li>• INDOUBT - APPLABND (abnormal program termination due to indoubt state)</li> <li>• INDOUBT - CANCEL (thread was in indoubt state when DB2 STOP FORCE/CANCEL was issued)</li> <li>• INDOUBT - EOT (DB2 completed the first phase of a 2-phase commit for the thread; the application terminated without properly disconnecting from DB2)</li> <li>• INDOUBT - RESOLVE (abnormal program termination while thread was in indoubt state)</li> <li>• NORMAL (normal thread termination)</li> <li>• PARTIAL (normal thread termination, but the thread was active before the monitor connected to DB2). The following statistics may be partial totals: <ul style="list-style-type: none"> <li>• plan information</li> <li>• thread transit times</li> <li>• sort statistics</li> <li>• commit statistics</li> <li>• thread resource locking and physical I/O counts</li> </ul> </li> <li>• RESOLVE INDOUBT (abnormal program termination due to recover indoubt)</li> <li>• SAME AUTH ID (a reusable thread that went through partial sign-on processing with the same authorization ID).</li> </ul>

Glossary Term or Monitor Field	Description (and Recommendations)
Accounting Class 2 Not Accounted Time	<p>The following simple formula defines DB2 Class 2 Not Accounted Time:</p> $\text{DB2 Class 2 Not Accounted Time} = \text{DB2 Class 2 Elapsed time} - (\text{DB2 Class 2 CPU time} + \text{DB2 Class 3 suspension time})$ <ul style="list-style-type: none"> <li>• Usually the DB2 Class 2 Not Accounted time is very small or negligible. It represents time that DB2 is unable to account for. If you see significant DB2 Class 2 Not Accounted time, it could be due to one of the following reasons:</li> <li>• In some cases, high DB2 Class 2 Not Accounted time is caused by too much detailed online tracing or bugs in some vendor performance monitors.</li> <li>• Reduce the level of tracing or stop the vendor performance monitor to help reduce the Not Accounted time to an acceptable level. This situation is usually the primary cause of high not-accounted-for-time on systems that are not CPU-constrained.</li> <li>• In a non-data sharing environment, it could be due to running on a very high CPU utilization environment and waiting for CPU cycles, especially with lower dispatching priority in Work Load Manager goal mode.</li> <li>• A non-dedicated LPAR can be interrupted by another LPAR and lose the processor for some time.</li> <li>• Also in a non-data sharing environment, it could be due to running in a high MVS paging environment and waiting for storage allocation. If DB2 gets swapped out by losing control of the processor or waiting for a processor, this increased time is the result.</li> <li>• In a data sharing environment, prior to V7, it could be due to asynchronous coupling facility requests. For example, group buffer pool requests for &gt; 4KB pages, long running coupling facility commands such as 'Read Directory Info' and 'Delete Name under mask', or conversion of synchronous requests to asynchronous requests due to a coupling facility subchannel busy condition. In V7 or later, the coupling facility suspensions due to asynchronous requests are shown under a new category called 'Asynch IXL Requests' in DB2 performance monitor. If the asynchronous write to the secondary group buffer pool (GBP) does not complete before synchronous write to the primary GBP with GBP duplexing, not accounted time can be the result.</li> <li>• Not accounted time can be the wait time for return from requests to be returned from VTAM or TCP/IP.</li> <li>• Instrumentation Facility Interface (IFI) log read can cause not accounted time.</li> <li>• If the environment is very I/O intensive, the Media Manager might be running out of request blocks.</li> <li>• z/OS events can cause not accounted time, such as SRM timer pops, for example when an SMF SRB is triggered to collect open data set statistics.</li> <li>• Waiting for package accounting can result in not accounted time.</li> <li>• Not accounted time can be accounted for by a PREPARE which is not</li> </ul>

Glossary Term or Monitor Field	Description (and Recommendations)
	<p>found in the Dynamic Statement Cache.</p> <ul style="list-style-type: none"> <li>• Waiting for a page being moved from VP to HP can result in not accounted time.</li> <li>• DD consolidation (z/OS parameter DDCONS=YES DETAIL) has overhead that is not accounted. See DDCONS informational APAR I107124.</li> <li>• Data set open contention related to PCLOSET being too small can cause time that is not accounted.</li> <li>• Time for RMF interval data set statistics gathering can cause not accounted time.</li> </ul> <p>DB2 internal suspend and resume can cause this not accounted time by looping when several threads are waiting for the same resource, but this case is very rare.</p> <p>For DB2 for z/OS Version 8, 9 and 10 see <a href="http://www-01.ibm.com/support/docview.wss?uid=swg21045823">http://www-01.ibm.com/support/docview.wss?uid=swg21045823</a></p>
EDM Pool Full	<p>This counter is incremented when an operation cannot proceed because the EDM pool was full. This is an extremely undesirable situation in which all pages in the Environmental Descriptor Manager pool (EDM pool) are allocated and in use as database descriptors (DBDs), skeleton cursor tables (SKCTs, which are internal copies of plans), skeleton package tables (SKPTs, which are internal copies of packages), and working cursor tables (CTs), and working package tables (PTs). The CTs and PTs are non-stealable while in use. No other operation can be commenced until one or more pages are freed in this pool.</p> <p>This number should be as close to zero as possible. If this situation occurs, consider increasing the number of pages in the EDM pool in DSNZPARM, on installation panel DSNTIPC.</p>
Transaction dispatch time and CPU time	 <p>The transaction total dispatch time is the total elapsed time during which the user task was dispatched by the CICS dispatcher domain on each CICS TCB under which the task executed.</p> <p>The transaction total CPU time is the total processor time during which the user task was dispatched by the CICS dispatcher domain on each CICS TCB under which the task executed.</p>

Glossary Term or Monitor Field	Description (and Recommendations)
--------------------------------	-----------------------------------

Wait (suspend) time relationships



The picture shows the relationship between a typical transaction wait time, and the transaction's suspend time, dispatch time, CPU and dispatch wait time.

It contains the elapsed time spent waiting for particular types of I/O operations. For example elapsed time waiting for terminal I/O. The elapsed time includes not only that time during which the I/O operation is actually taking place, but also the time during which the access method is completing the outstanding event control block, and the time subsequent to that until the waiting CICS transaction is redispached.