







**Database Monitoring Guide and Reference**

**Note**

Before using this information and the product it supports, read the general information under Appendix B, "Notices," on page 763.

**Edition Notice**

This document contains proprietary information of IBM. It is provided under a license agreement and is protected by copyright law. The information contained in this publication does not include any product warranties, and any statements provided in this manual should not be interpreted as such.

You can order IBM publications online or through your local IBM representative.

- To order publications online, go to the IBM Publications Center at [www.ibm.com/shop/publications/order](http://www.ibm.com/shop/publications/order)
- To find your local IBM representative, go to the IBM Directory of Worldwide Contacts at [www.ibm.com/planetwide](http://www.ibm.com/planetwide)

To order DB2 publications from DB2 Marketing and Sales in the United States or Canada, call 1-800-IBM-4YOU (426-4968).

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 1993, 2009.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

---

# Contents

**About this book . . . . . xvii**

---

**Part 1. Monitoring interfaces. . . . . 1**

**Chapter 1. Database monitoring . . . . . 3**

**Chapter 2. Monitor table functions overview . . . . . 5**

Monitoring system information using table functions . . . . . 5  
Monitoring activities using table functions . . . . . 6  
Monitoring data objects using table functions . . . . . 7

**Chapter 3. Event monitors . . . . . 9**

Event monitors that write to an unformatted event table. . . . . 11  
    Unformatted event table column definitions . . . . . 13  
    db2evmonfmt tool for reading event monitor data . . . . . 16  
    Monitoring database locking. . . . . 21  
    Monitoring unit of work events . . . . . 41  
    Capturing system monitor elements using the statistics event monitor . . . . . 49  
    Capturing activity monitor elements using the activity event monitor . . . . . 54  
Event monitors that write to tables, files, and pipes . . . . . 58  
    Collecting information about database system events . . . . . 58  
    Creating an event monitor . . . . . 60  
    Event monitor sample output . . . . . 75

**Chapter 4. Snapshot monitor . . . . . 83**

Access to system monitor data: SYSMON authority . . . . . 83  
Capturing database system snapshots using snapshot administrative views and table functions . . . . . 84  
Capturing database system snapshot information to a file using the SNAP\_WRITE\_FILE stored procedure . . . . . 86  
Accessing database system snapshots using snapshot table functions in SQL queries (with file access) . . . . . 88  
Snapshot monitor SQL Administrative Views . . . . . 89  
SQL access to database system snapshots . . . . . 92  
Capturing a database snapshot from the CLP . . . . . 93  
Snapshot monitor CLP commands. . . . . 93  
Capturing a database snapshot from a client application . . . . . 96  
Snapshot monitor API request types . . . . . 97  
Snapshot monitor sample output . . . . . 99  
Subsection snapshots . . . . . 101  
Global snapshots on partitioned database systems . . . . . 102  
Snapshot monitor self-describing data stream. . . . . 103  
Monitoring with db2top in interactive mode commands . . . . . 105  
    .db2toprc configuration file. . . . . 108

**Chapter 5. Switch-based monitoring concepts . . . . . 111**

System monitor switches . . . . . 111  
    Setting system monitor switches from the CLP . . . . . 113  
    Setting system monitor switches from a client application . . . . . 115  
    System monitor switches self-describing data stream. . . . . 116  
Database system monitor data organization . . . . . 117  
Counter status and visibility . . . . . 118  
System monitor output: the self-describing data stream. . . . . 119  
Memory requirements for monitor data. . . . . 119  
Monitoring buffer pool activity . . . . . 122  
Database system monitor interfaces . . . . . 124

**Chapter 6. Deprecated monitoring tools . . . . . 127**

Health monitor. . . . . 127  
    Monitoring database health. . . . . 127  
    Health indicators . . . . . 159  
Working with the Memory Visualizer . . . . . 185  
    Memory Visualizer overview . . . . . 187  
Activity Monitor overview . . . . . 190  
    Monitoring scenarios . . . . . 193  
    Setting up an activity monitor . . . . . 196  
    Progress monitoring of the rollback process . . . . . 196  
    Using snapshot monitor data to monitor the reorganization of a partitioned table. . . . . 197  
    Inactive statement tracking for DEADLOCK WITH DETAILS HISTORY event monitors. . . . . 205  
Introduction to Windows Management Instrumentation (WMI) . . . . . 206  
    DB2 database system integration with Windows Management Instrumentation . . . . . 207  
    Windows performance monitor introduction . . . . . 208  
Indoubt Transaction Manager overview. . . . . 211

---

**Part 2. Monitor elements . . . . . 215**

**Chapter 7. Monitor elements reported in monitor table functions . . . . . 217**

**Chapter 8. Request monitor elements . . . . . 227**

**Chapter 9. Activity monitor elements . . . . . 229**

**Chapter 10. Data object monitor elements . . . . . 231**

**Chapter 11. Monitor elements reported by the unit of work event monitor . . . . . 233**

**Chapter 12. Monitor elements reported by the locking event monitor. . . . . 235**

**Chapter 13. Wait time monitor elements . . . . . 237**

**Chapter 14. Logical data groups . . . . . 241**

Snapshot monitor interface mappings to logical data groups . . . . . 241  
 Snapshot monitor logical data groups and monitor elements . . . . . 245  
 Event type mappings to logical data groups . . . . . 276  
 Event monitor logical data groups and monitor elements . . . . . 279  
 Logical data groups affected by COLLECT ACTIVITY DATA settings . . . . . 300

**Chapter 15. Database system monitor elements . . . . . 301**

acc\_curs\_blk - Accepted Block Cursor Requests . . . . . 302  
 act\_aborted\_total - Total aborted activities monitor element . . . . . 302  
 act\_completed\_total - Total completed activities monitor element . . . . . 303  
 act\_cpu\_time\_top - Activity CPU time top monitor element . . . . . 304  
 act\_exec\_time - Activity execution time monitor element . . . . . 305  
 act\_rejected\_total - Total rejected activities monitor element . . . . . 305  
 act\_remapped\_in - Activities remapped in monitor element . . . . . 306  
 act\_remapped\_out - Activities remapped out monitor element . . . . . 306  
 act\_rows\_read\_top - Activity rows read top monitor element . . . . . 307  
 act\_total - Activities total monitor element. . . . . 307  
 activate\_timestamp - Activate timestamp monitor element . . . . . 308  
 active\_hash\_joins - Active hash joins . . . . . 308  
 active\_olap\_funcs - Active OLAP Functions monitor element . . . . . 308  
 active\_sorts - Active Sorts . . . . . 308  
 activity\_collected - Activity collected monitor element . . . . . 309  
 activity\_id - Activity ID monitor element . . . . . 309  
 activity\_secondary\_id - Activity secondary ID monitor element . . . . . 310  
 activity\_state - Activity state monitor element . . . . . 310  
 activity\_type - Activity type monitor element. . . . . 311  
 activitytotaltime\_threshold\_id - Activity total time threshold ID monitor element . . . . . 311  
 activitytotaltime\_threshold\_value - Activity total time threshold value monitor element . . . . . 312  
 activitytotaltime\_threshold\_violated - Activity total time threshold violated monitor element . . . . . 312  
 address - IP address from which the connection was initiated . . . . . 312

agent\_id - Application handle (agent ID) monitor element . . . . . 313  
 agent\_id\_holding\_lock - Agent ID Holding Lock 314  
 agent\_pid - Engine dispatchable unit (EDU) identifier monitor element . . . . . 315  
 agent\_status - DCS Application Agents. . . . . 315  
 agent\_sys\_cpu\_time - System CPU Time used by Agent . . . . . 315  
 agent\_usr\_cpu\_time - User CPU Time used by Agent . . . . . 316  
 agent\_wait\_time - Agent wait time monitor element . . . . . 316  
 agent\_waits\_total - Total agent waits monitor element . . . . . 318  
 agents\_created\_empty\_pool - Agents Created Due to Empty Agent Pool. . . . . 319  
 agents\_from\_pool - Agents Assigned From Pool 319  
 agents\_registered - Agents Registered . . . . . 319  
 agents\_registered\_top - Maximum Number of Agents Registered . . . . . 320  
 agents\_stolen - Stolen Agents . . . . . 320  
 agents\_top - Number of Agents Created . . . . . 321  
 agents\_waiting\_on\_token - Agents Waiting for a Token . . . . . 321  
 agents\_waiting\_top - Maximum Number of Agents Waiting monitor element . . . . . 321  
 agg\_temp\_tablespace\_top - Aggregate temporary table space top monitor element . . . . . 322  
 aggsqtempespace\_threshold\_id - Aggregate SQL temporary space threshold ID monitor element . . . . . 322  
 aggsqtempespace\_threshold\_value - AggSQL temporary space threshold value monitor element . . . . . 323  
 aggsqtempespace\_threshold\_violated - AggSQL temporary space threshold violated monitor element . . . . . 323  
 app\_rqsts\_completed\_total - Total application requests completed monitor element. . . . . 323  
 appl\_con\_time - Connection Request Start Timestamp . . . . . 324  
 appl\_id - Application ID. . . . . 325  
 appl\_id\_holding\_lk - Application ID Holding Lock 327  
 appl\_id\_oldest\_xact - Application with Oldest Transaction . . . . . 327  
 appl\_idle\_time - Application Idle Time . . . . . 328  
 appl\_name - Application name monitor element 328  
 appl\_priority - Application Agent Priority. . . . . 329  
 appl\_priority\_type - Application Priority Type . . . . . 329  
 appl\_section\_inserts - Section Inserts monitor element . . . . . 330  
 appl\_section\_lookups - Section Lookups . . . . . 330  
 appl\_status - Application Status . . . . . 331  
 application\_handle - Application handle monitor element . . . . . 333  
 appls\_cur\_cons - Applications Connected Currently 334  
 appls\_in\_db2 - Applications Executing in the Database Currently . . . . . 334  
 arm\_correlator - Application response measurement correlator monitor element . . . . . 335  
 associated\_agents\_top - Maximum Number of Associated Agents. . . . . 335

async_runstats - Total number of asynchronous RUNSTATS requests monitor element . . . . .	335	client_prdid - Client product and version ID monitor element . . . . .	357
audit_events_total - Total audit events monitor element . . . . .	336	client_protocol - Client Communication Protocol	358
audit_file_write_wait_time - Audit file write wait time monitor element. . . . .	336	client_userid - Client user ID monitor element . . . . .	358
audit_file_writes_total - Total audit files written monitor element . . . . .	337	client_wrkstnname - Client workstation name monitor element . . . . .	359
audit_subsystem_wait_time - Audit subsystem wait time monitor element. . . . .	338	codepage_id - ID of Code Page Used by Application . . . . .	360
audit_subsystem_waits_total - Total audit subsystem waits monitor element . . . . .	340	comm_private_mem - Committed Private Memory	361
auth_id - Authorization ID . . . . .	340	commit_sql_stmts - Commit Statements Attempted	361
authority_bitmap - User authorization level monitor element . . . . .	341	comp_env_desc - Compilation environment monitor element . . . . .	362
authority_lvl - User authorization level monitor element . . . . .	342	completion_status - Completion status monitor element . . . . .	362
auto_storage_hybrid - Hybrid automatic storage table space indicator monitor element . . . . .	343	con_elapsed_time - Most Recent Connection Elapsed Time . . . . .	362
automatic - Buffer pool automatic monitor element	343	con_local_databases - Local Databases with Current Connects . . . . .	363
bin_id - Histogram bin identifier monitor element	343	con_response_time - Most Recent Response Time for Connect . . . . .	363
binds_precompiles - Binds/Precompiles Attempted	344	concurrent_act_top - Concurrent activity top monitor element . . . . .	363
block_ios - Number of block I/O requests monitor element . . . . .	344	concurrent_connection_top - Concurrent connection top monitor element . . . . .	364
blocking_cursor - Blocking Cursor . . . . .	345	concurrent_wlo_act_top - Concurrent WLO activity top monitor element . . . . .	364
blocks_pending_cleanup - Pending cleanup rolled-out blocks monitor element . . . . .	346	concurrent_wlo_top - Concurrent workload occurrences top monitor element . . . . .	364
bottom - Histogram bin bottom monitor element	346	concurrentdbcoordactivities_db_threshold_id - Concurrent database coordinator activities database threshold ID monitor element . . . . .	365
boundary_leaf_node_splits - Boundary leaf node splits monitor element . . . . .	346	concurrentdbcoordactivities_db_threshold_queued - Concurrent database coordinator activities database threshold queued monitor element . . . . .	365
bp_cur_buffsz - Current Size of Buffer Pool . . . . .	346	concurrentdbcoordactivities_db_threshold_value - Concurrent database coordinator activities database threshold value monitor element . . . . .	365
bp_id - Buffer pool identifier monitor element . . . . .	347	concurrentdbcoordactivities_db_threshold_violated - Concurrent database coordinator activities database threshold violated monitor element . . . . .	366
bp_name - Buffer pool name monitor element . . . . .	347	concurrentdbcoordactivities_subclass_threshold_id - Concurrent database coordinator activities service subclass threshold ID monitor element . . . . .	366
bp_new_buffsz - New Buffer Pool Size . . . . .	347	concurrentdbcoordactivities_subclass_threshold_queued - Concurrent database coordinator activities service subclass threshold queued monitor element . . . . .	367
bp_pages_left_to_remove - Number of Pages Left to Remove . . . . .	348	concurrentdbcoordactivities_subclass_threshold_value - Concurrent database coordinator activities service subclass threshold value monitor element . . . . .	367
bp_tbsp_use_count - Number of Table Spaces Mapped to Buffer Pool . . . . .	348	concurrentdbcoordactivities_subclass_threshold_violated - Concurrent database coordinator activities service subclass threshold violated monitor element . . . . .	367
buff_free - FCM Buffers Currently Free . . . . .	348	concurrentdbcoordactivities_superclass_threshold_id - Concurrent database coordinator activities service superclass threshold ID monitor element . . . . .	368
buff_free_bottom - Minimum FCM Buffers Free	348		
byte_order - Byte Order of Event Data . . . . .	349		
cat_cache_inserts - Catalog Cache Inserts . . . . .	349		
cat_cache_lookups - Catalog Cache Lookups . . . . .	350		
cat_cache_overflows - Catalog Cache Overflows	351		
cat_cache_size_top - Catalog cache high watermark monitor element . . . . .	351		
catalog_node - Catalog Node Number . . . . .	352		
catalog_node_name - Catalog Node Network Name . . . . .	352		
ch_free - Channels Currently Free . . . . .	353		
ch_free_bottom - Minimum Channels Free. . . . .	353		
client_actng - Client accounting string monitor element . . . . .	353		
client_applname - Client application name monitor element . . . . .	354		
client_db_alias - Database Alias Used by Application . . . . .	355		
client_idle_wait_time - Client idle wait time monitor element . . . . .	355		
client_pid - Client Process ID . . . . .	356		
client_platform - Client Operating Platform . . . . .	357		



concurrentdbcoordactivities_superclass_		coord_act_lifetime_top - Coordinator activity	
threshold_queued - Concurrent database		lifetime top monitor element . . . . .	380
coordinator activities service superclass threshold		coord_member - Coordinator member monitor	
queued monitor element . . . . .	368	element . . . . .	380
concurrentdbcoordactivities_superclass_		coord_act_queue_time_avg - Coordinator activity	
threshold_value - Concurrent database coordinator		queue time average monitor element . . . . .	381
activities service superclass threshold value		coord_act_rejected_total - Coordinator activities	
monitor element . . . . .	369	rejected total monitor element . . . . .	382
concurrentdbcoordactivities_superclass_		coord_agent_pid - Coordinator agent identifier	
threshold_violated - Concurrent database		monitor element . . . . .	382
coordinator activities service superclass threshold		coord_agents_top - Maximum Number of	
violated monitor element . . . . .	369	Coordinating Agents . . . . .	382
concurrentdbcoordactivities_work_action_set_		coord_node - Coordinating Node. . . . .	383
threshold_id - Concurrent database coordinator		coord_partition_num - Coordinator partition	
activities work action set threshold ID monitor		number monitor element . . . . .	383
element . . . . .	369	corr_token - DRDA Correlation Token . . . . .	383
concurrentdbcoordactivities_work_action_set_		cost_estimate_top - Cost estimate top monitor	
threshold_queued - Concurrent database		element . . . . .	384
coordinator activities work action set threshold		count - Number of Event Monitor Overflows. . . . .	384
queued monitor element . . . . .	370	cptime_threshold_id - CPU time threshold ID	
concurrentdbcoordactivities_work_action_set_		monitor element . . . . .	385
threshold_value - Concurrent database coordinator		cptime_threshold_value - CPU time threshold	
activities work action set threshold value monitor		value monitor element . . . . .	385
element . . . . .	370	cptime_threshold_violated - CPU time threshold	
concurrentdbcoordactivities_work_action_set_		violated monitor element . . . . .	385
threshold_violated - Concurrent database		cptimeinsc_threshold_id - CPU time in service	
coordinator activities work action set threshold		class threshold ID monitor element . . . . .	386
violated monitor element . . . . .	371	cptimeinsc_threshold_value - CPU time in service	
conn_complete_time - Connection Request		class threshold value monitor element . . . . .	386
Completion Timestamp . . . . .	371	cptimeinsc_threshold_violated - CPU time in	
conn_time - Time of database connection monitor		service class threshold violated monitor element. . . . .	386
element . . . . .	371	create_nickname - Create Nicknames . . . . .	387
connection_status - Connection Status . . . . .	372	create_nickname_time - Create Nickname Response	
connections_top - Maximum Number of		Time . . . . .	387
Concurrent Connections. . . . .	372	creator - Application Creator . . . . .	387
consistency_token - Package consistency token		current_active_log - Current Active Log File	
monitor element . . . . .	373	Number . . . . .	388
container_accessible - Accessibility of container		current_archive_log - Current Archive Log File	
monitor element . . . . .	373	Number . . . . .	388
container_id - Container identification monitor		current_extent - Extent currently being moved	
element . . . . .	373	monitor element . . . . .	389
container_name - Container name monitor element		cursor_name - Cursor Name . . . . .	389
374		data_object_pages - Data Object Pages . . . . .	390
container_stripe_set - Container stripe set monitor		data_partition_id - Data partition identifier monitor	
element . . . . .	374	element . . . . .	390
container_total_pages - Total pages in container		datasource_name - Data Source Name . . . . .	391
monitor element . . . . .	375	db2_status - Status of DB2 Instance . . . . .	391
container_type - Container type monitor element		db2start_time - Start Database Manager Timestamp	392
375		db_conn_time - Database activation timestamp	
container_usable_pages - Usable pages in container		monitor element . . . . .	392
monitor element . . . . .	375	db_heap_top - Maximum Database Heap Allocated	392
coord_act_aborted_total - Coordinator activities		db_location - Database Location . . . . .	393
aborted total monitor element . . . . .	376	db_name - Database Name. . . . .	393
coord_act_completed_total - Coordinator activities		db_path - Database Path. . . . .	394
completed total monitor element . . . . .	376	db_status - Status of Database. . . . .	394
coord_act_est_cost_avg - Coordinator activity		db_storage_path - Automatic storage path monitor	
estimated cost average monitor element . . . . .	377	element . . . . .	395
coord_act_exec_time_avg - Coordinator activities		db_storage_path_state - Storage path state monitor	
execution time average monitor element . . . . .	378	element . . . . .	395
coord_act_interarrival_time_avg - Coordinator			
activity arrival time average monitor element . . . . .	378		
coord_act_lifetime_avg - Coordinator activity			
lifetime average monitor element. . . . .	379		



db_storage_path_with_dpe - Storage path including database partition expression monitor element . . . . .	395	estimatedsqlcost_threshold_value - Estimated SQL cost threshold value monitor element . . . . .	418
db_work_action_set_id - Database work action set ID monitor element . . . . .	396	estimatedsqlcost_threshold_violated - Estimated SQL cost threshold violated monitor element . . . . .	418
db_work_class_id - Database work class ID monitor element . . . . .	396	event_monitor_name - Event Monitor Name . . . . .	419
dcs_appl_status - DCS Application Status . . . . .	397	event_time - Event Time. . . . .	419
dcs_db_name - DCS Database Name . . . . .	397	evmon_activates - Number of Event Monitor Activations . . . . .	419
ddl_sql_stmts - Data Definition Language (DDL) SQL Statements. . . . .	397	executable_id - Executable ID monitor element . . . . .	420
deadlock_id - Deadlock Event Identifier . . . . .	398	evmon_flushes - Number of Event Monitor Flushes . . . . .	420
deadlock_node - Partition Number Where Deadlock Occurred . . . . .	399	execution_id - User Login ID . . . . .	421
deadlocks - Deadlocks detected monitor element . . . . .	399	failed_sql_stmts - Failed Statement Operations . . . . .	421
degree_parallelism - Degree of Parallelism. . . . .	401	fcm_message_rcv_volume - FCM message received volume monitor element . . . . .	422
del_keys_cleaned - Pseudo deleted keys cleaned monitor element . . . . .	401	fcm_message_rcv_wait_time - FCM message received wait time monitor element . . . . .	423
delete_sql_stmts - Deletes . . . . .	401	fcm_message_rcvts_total - Total FCM message receives monitor element . . . . .	424
delete_time - Delete Response Time . . . . .	402	fcm_message_send_volume - FCM message send volume monitor element . . . . .	424
destination_service_class_id - Destination service class ID monitor element . . . . .	402	fcm_message_send_wait_time - FCM message send wait time monitor element . . . . .	425
diaglog_write_wait_time - Diagnostic log file write wait time monitor element . . . . .	402	fcm_message_sends_total - Total FCM message sends monitor element . . . . .	426
diaglog_writes_total - Total diagnostic log file writes monitor element . . . . .	403	fcm_rcv_volume - FCM received volume monitor element . . . . .	426
direct_read_reqs - Direct read requests monitor element . . . . .	404	fcm_rcv_wait_time - FCM received wait time monitor element . . . . .	427
direct_read_time - Direct read time monitor element . . . . .	406	fcm_rcvts_total - FCM receives total monitor element . . . . .	429
direct_reads - Direct reads from database monitor element . . . . .	407	fcm_send_volume - FCM send volume monitor element . . . . .	430
direct_write_reqs - Direct write requests monitor element . . . . .	409	fcm_send_wait_time - FCM send wait time monitor element . . . . .	430
direct_write_time - Direct write time monitor element . . . . .	410	fcm_sends_total - FCM sends total monitor element . . . . .	431
direct_writes - Direct writes to database monitor element . . . . .	412	fcm_tq_rcv_volume - FCM table queue received volume monitor element . . . . .	433
disconn_time - Database Deactivation Timestamp . . . . .	413	fcm_tq_rcv_wait_time - FCM table queue received wait time monitor element . . . . .	433
disconnects - Disconnects . . . . .	414	fcm_tq_rcvts_total - FCM table queue receives total monitor element . . . . .	434
dl_conns - Connections involved in deadlock monitor element . . . . .	414	fcm_tq_send_volume - FCM table queue send volume monitor element . . . . .	435
dynamic_sql_stmts - Dynamic SQL Statements Attempted . . . . .	414	fcm_tq_send_wait_time - FCM table queue send wait time monitor element . . . . .	436
eff_stmt_text - Effective statement text monitor element . . . . .	415	fcm_tq_sends_total - FCM table queue send total monitor element . . . . .	437
effective_isolation - Effective isolation monitor element . . . . .	415	fetch_count - Number of Successful Fetches . . . . .	438
effective_lock_timeout - Effective lock timeout monitor element . . . . .	416	files_closed - Database files closed monitor element . . . . .	438
effective_query_degree - Effective query degree monitor element . . . . .	416	first_active_log - First Active Log File Number . . . . .	439
elapsed_exec_time - Statement Execution Elapsed Time . . . . .	416	first_overflow_time - Time of First Event Overflow . . . . .	440
empty_pages_deleted - Empty pages deleted monitor element . . . . .	417	fs_caching - File system caching monitor element . . . . .	440
empty_pages_reused - Empty pages reused monitor element . . . . .	417	fs_id - Unique file system identification number monitor element . . . . .	440
entry_time - Entry time monitor element . . . . .	417	fs_total_size - Total size of a file system monitor element . . . . .	441
estimatedsqlcost_threshold_id - Estimated SQL cost threshold ID monitor element . . . . .	418	fs_type - File System Type . . . . .	441
		fs_used_size - Amount of space used on a file system monitor element . . . . .	442
		gw_comm_error_time - Communication Error Time . . . . .	442
		gw_comm_errors - Communication Errors. . . . .	442

gw_con_time - DB2 Connect Gateway First Connect Initiated . . . . .	443	iid - Index identifier monitor element . . . . .	458
gw_connections_top - Maximum Number of Concurrent Connections to Host Database. . . . .	443	inbound_bytes_received - Inbound Number of Bytes Received . . . . .	459
gw_cons_wait_client - Number of Connections Waiting for the Client to Send Request . . . . .	443	inbound_bytes_sent - Inbound Number of Bytes Sent . . . . .	459
gw_cons_wait_host - Number of Connections Waiting for the Host to Reply . . . . .	444	inbound_comm_address - Inbound Communication Address . . . . .	459
gw_cur_cons - Current Number of Connections for DB2 Connect . . . . .	444	include_col_updates - Include column updates monitor element . . . . .	460
gw_db_alias - Database Alias at the Gateway. . . . .	444	index_object_pages - Index Object Pages . . . . .	460
gw_exec_time - Elapsed Time Spent on DB2 Connect Gateway Processing . . . . .	445	index_only_scans - Index-only scans monitor element . . . . .	460
gw_total_cons - Total Number of Attempted Connections for DB2 Connect . . . . .	445	index_scans - Index scans monitor element . . . . .	460
hadr_connect_status - HADR Connection Status monitor element . . . . .	445	index_tbsp_id - Index table space ID monitor element . . . . .	460
hadr_connect_time - HADR Connection Time monitor element . . . . .	446	input_db_alias - Input Database Alias . . . . .	461
hadr_heartbeat - HADR Heartbeat monitor element	447	insert_sql_stmts - Inserts . . . . .	461
hadr_local_host - HADR Local Host monitor element . . . . .	447	insert_time - Insert Response Time . . . . .	462
hadr_local_service - HADR Local Service monitor element . . . . .	448	insert_timestamp - Statement insert timestamp monitor element . . . . .	462
hadr_log_gap - HADR Log Gap . . . . .	448	int_auto_rebinds - Internal Automatic Rebinds . . . . .	463
hadr_peer_window - HADR peer window monitor element . . . . .	449	int_commits - Internal Commits . . . . .	463
hadr_peer_window_end - HADR peer window end monitor element . . . . .	449	int_deadlock_rollbacks - Internal Rollbacks Due To Deadlock . . . . .	464
hadr_primary_log_file - HADR Primary Log File monitor element . . . . .	449	int_node_splits - Intermediate node splits monitor element . . . . .	465
hadr_primary_log_lsn - HADR Primary Log LSN monitor element . . . . .	450	int_rollbacks - Internal Rollbacks . . . . .	465
hadr_primary_log_page - HADR Primary Log Page monitor element . . . . .	450	int_rows_deleted - Internal Rows Deleted . . . . .	466
hadr_remote_host - HADR Remote Host monitor element . . . . .	450	int_rows_inserted - Internal Rows Inserted . . . . .	466
hadr_remote_instance - HADR Remote Instance monitor element . . . . .	451	int_rows_updated - Internal Rows Updated . . . . .	467
hadr_remote_service - HADR Remote Service monitor element . . . . .	451	invocation_id - Invocation ID monitor element . . . . .	468
hadr_role - HADR Role . . . . .	452	ipc_rcv_volume - Interprocess communication received volume monitor element . . . . .	468
hadr_standby_log_file - HADR Standby Log File monitor element . . . . .	452	ipc_rcv_wait_time - Interprocess communication received wait time monitor element . . . . .	469
hadr_standby_log_lsn - HADR Standby Log LSN monitor element . . . . .	452	ipc_rcvcs_total - Interprocess communication receives total monitor element . . . . .	469
hadr_standby_log_page - HADR Standby Log Page monitor element . . . . .	453	ipc_send_volume - Interprocess communication send volume monitor element . . . . .	470
hadr_state - HADR State monitor element . . . . .	453	ipc_send_wait_time - Interprocess communication send wait time monitor element . . . . .	471
hadr_syncmode - HADR Synchronization Mode monitor element . . . . .	454	ipc_sends_total - Interprocess communication send total monitor element . . . . .	472
hadr_timeout - HADR Timeout monitor element	454	is_system_appl - Is System Application monitor element . . . . .	473
hash_join_overflows - Hash Join Overflows . . . . .	455	key_updates - Key updates monitor element . . . . .	473
hash_join_small_overflows - Hash Join Small Overflows . . . . .	455	last_active_log - Last Active Log File Number . . . . .	473
histogram_type - Histogram type monitor element	456	last_backup - Last Backup Timestamp . . . . .	474
host_ccsid - Host Coded Character Set ID . . . . .	457	last_extent - Last extent moved monitor element	474
host_db_name - Host Database Name . . . . .	457	last_overflow_time - Time of Last Event Overflow	474
host_prdid - Host Product/Version ID . . . . .	457	last_reference_time - Last reference time monitor element . . . . .	475
host_response_time - Host Response Time. . . . .	458	last_reset - Last Reset Timestamp. . . . .	475
idle_agents - Number of Idle Agents . . . . .	458	last_wlm_reset - Time of last reset monitor element	475
		lob_object_pages - LOB Object Pages . . . . .	476
		local_cons - Local Connections . . . . .	476
		local_cons_in_exec - Local Connections Executing in the Database Manager . . . . .	477
		local_start_time - Local start time monitor element	477
		lock_attributes - Lock attributes monitor element	477
		lock_count - Lock count monitor element . . . . .	478

lock_current_mode - Original Lock Mode Before Conversion . . . . .	479	max_data_received_2048 - Number of Statements with Outbound Bytes Received Between 1025 and 2048 Bytes . . . . .	503
lock_escalation - Lock escalation monitor element . . . . .	480	max_data_received_256 - Number of Statements with Outbound Bytes Received Between 129 and 256 Bytes . . . . .	504
lock_escals - Number of lock escalations monitor element . . . . .	480	max_data_received_31999 - Number of Statements with Outbound Bytes Received Between 16385 and 31999 Bytes monitor element . . . . .	504
lock_hold_count - Lock hold count monitor element . . . . .	482	max_data_received_4096 - Number of Statements with Outbound Bytes Received Between 2049 and 4096 Bytes . . . . .	505
lock_list_in_use - Total Lock List Memory In Use . . . . .	482	max_data_received_512 - Number of Statements with Outbound Bytes Received Between 257 and 512 Bytes . . . . .	505
lock_mode - Lock mode monitor element . . . . .	483	max_data_received_64000 - Number of Statements with Outbound Bytes Received Between 32000 and 64000 Bytes monitor element . . . . .	505
lock_mode_requested - Lock mode requested monitor element . . . . .	484	max_data_received_8192 - Number of Statements with Outbound Bytes Received Between 4097 and 8192 Bytes . . . . .	506
lock_name - Lock name monitor element . . . . .	484	max_data_received_gt64000 - Number of Statements with Outbound Bytes Received Greater than 64000 Bytes . . . . .	506
lock_node - Lock Node . . . . .	485	max_data_sent_1024 - Number of Statements with Outbound Bytes Sent Between 513 and 1024 Bytes . . . . .	507
lock_object_name - Lock Object Name . . . . .	485	max_data_sent_128 - Number of Statements with Outbound Bytes Sent Between 1 and 128 Bytes . . . . .	507
lock_object_type - Lock object type waited on monitor element . . . . .	486	max_data_sent_16384 - Number of Statements with Outbound Bytes Sent Between 8193 and 16384 Bytes . . . . .	508
lock_release_flags - Lock release flags monitor element . . . . .	486	max_data_sent_2048 - Number of Statements with Outbound Bytes Sent Between 1025 and 2048 Bytes . . . . .	508
lock_status - Lock status monitor element . . . . .	487	max_data_sent_256 - Number of Statements with Outbound Bytes Sent Between 129 and 256 Bytes . . . . .	508
lock_timeout_val - Lock timeout value monitor element . . . . .	488	max_data_sent_31999 - Number of Statements with Outbound Bytes Sent Between 16385 and 31999 Bytes . . . . .	509
lock_timeouts - Number of lock timeouts monitor element . . . . .	488	max_data_sent_4096 - Number of Statements with Outbound Bytes Sent Between 2049 and 4096 Bytes . . . . .	509
lock_wait_start_time - Lock Wait Start Timestamp . . . . .	490	max_data_sent_512 - Number of Statements with Outbound Bytes Sent Between 257 and 512 Bytes . . . . .	510
lock_wait_time - Time waited on locks monitor element . . . . .	490	max_data_sent_64000 - Number of Statements with Outbound Bytes Sent Between 32000 and 64000 Bytes . . . . .	510
lock_wait_time_top - Lock wait time top monitor element . . . . .	492	max_data_sent_8192 - Number of Statements with Outbound Bytes Sent Between 4097 and 8192 Bytes . . . . .	511
lock_waits - Lock waits monitor element . . . . .	492	max_data_sent_gt64000 - Number of Statements with Outbound Bytes Sent Greater than 64000 Bytes . . . . .	511
locks_held - Locks Held . . . . .	494	max_network_time_100_ms - Number of Statements with Network Time between 16 and 100 ms . . . . .	512
locks_held_top - Maximum Number of Locks Held . . . . .	494	max_network_time_16_ms - Number of Statements with Network Time between 4 and 16 ms . . . . .	512
locks_in_list - Number of Locks Reported . . . . .	495	max_network_time_1_ms - Number of Statements with Network Time of up to 1 ms . . . . .	512
locks_waiting - Current Agents Waiting On Locks . . . . .	495	max_network_time_4_ms - Number of Statements with Network Time between 1 and 4 ms . . . . .	513
log_buffer_wait_time - Log buffer wait time monitor element . . . . .	495		
log_disk_wait_time - Log disk wait time monitor element . . . . .	496		
log_disk_waits_total - Total log disk waits monitor element . . . . .	497		
log_held_by_dirty_pages - Amount of Log Space Accounted for by Dirty Pages . . . . .	498		
log_read_time - Log Read Time . . . . .	499		
log_reads - Number of Log Pages Read . . . . .	499		
log_to_redo_for_recovery - Amount of Log to be Redone for Recovery . . . . .	500		
log_write_time - Log Write Time . . . . .	500		
log_writes - Number of Log Pages Written . . . . .	500		
long_object_pages - Long Object Pages . . . . .	501		
long_tbsp_id - Long table space ID monitor element . . . . .	501		
max_agent_overflows - Maximum Agent Overflows . . . . .	502		
max_data_received_1024 - Number of Statements with Outbound Bytes Received Between 513 and 1024 Bytes . . . . .	502		
max_data_received_128 - Number of Statements with Outbound Bytes Received Between 1 and 128 Bytes . . . . .	503		
max_data_received_16384 - Number of Statements with Outbound Bytes Received Between 8193 and 16384 Bytes . . . . .	503		

max_network_time_500_ms - Number of Statements with Network Time between 100 and 500 ms . . . . .	513	outbound_appl_id - Outbound Application ID . . . . .	529
max_network_time_gt500_ms - Number of Statements with Network Time greater than 500 ms	514	outbound_bytes_received - Outbound Number of Bytes Received . . . . .	530
member - Database member monitor element . . . . .	514	outbound_bytes_received_bottom - Minimum Outbound Number of Bytes Received . . . . .	531
message - Control Table Message . . . . .	515	outbound_bytes_received_top - Maximum Outbound Number of Bytes Received . . . . .	531
message_time - Timestamp Control Table Message	516	outbound_bytes_sent - Outbound Number of Bytes Sent . . . . .	531
nesting_level - Nesting level monitor element . . . . .	516	outbound_bytes_sent_bottom - Minimum Outbound Number of Bytes Sent . . . . .	532
network_time_bottom - Minimum Network Time for Statement . . . . .	516	outbound_bytes_sent_top - Maximum Outbound Number of Bytes Sent . . . . .	532
network_time_top - Maximum Network Time for Statement . . . . .	517	outbound_comm_address - Outbound Communication Address . . . . .	532
nleaf - Number of leaf pages monitor element . . . . .	517	outbound_comm_protocol - Outbound Communication Protocol . . . . .	533
nlevels - Number of index levels monitor element	517	outbound_sequence_no - Outbound Sequence Number . . . . .	533
node_number - Node Number . . . . .	517	overflow_accesses - Accesses to overflowed records monitor element . . . . .	533
nonboundary_leaf_node_splits - Non-boundary leaf node splits monitor element . . . . .	518	overflow_creates - Overflow creates monitor element . . . . .	534
num_agents - Number of Agents Working on a Statement . . . . .	518	package_name - Package name monitor element	534
num_assoc_agents - Number of Associated Agents	519	package_schema - Package schema monitor element . . . . .	535
num_compilations - Statement Compilations . . . . .	519	package_version_id - Package version monitor element . . . . .	535
num_db_storage_paths - Number of automatic storage paths . . . . .	519	page_allocations - Page allocations monitor element . . . . .	536
num_executions - Statement executions monitor element . . . . .	520	page_reorgs - Page Reorganizations . . . . .	536
num_exec_with_metrics - Number of executions with metrics collected monitor element. . . . .	520	pages_from_block_ios - Total number of pages read by block I/O monitor element. . . . .	536
num_extents_left - Number of extents left to process monitor element. . . . .	520	pages_from_vectored_ios - Total number of pages read by vectored I/O monitor element . . . . .	537
num_extents_moved - Number of extents moved monitor element . . . . .	521	pages_merged - Pages merged monitor element	538
num_gw_conn_switches - Connection Switches . . . . .	521	pages_read - Number of pages read monitor element . . . . .	538
num_indoubt_trans - Number of Indoubt Transactions . . . . .	521	pages_written - Number of pages written monitor element . . . . .	538
num_log_buffer_full - Number of full log buffers monitor element . . . . .	522	parent_activity_id - Parent activity ID monitor element . . . . .	538
num_log_data_found_in_buffer - Number of Log Data Found In Buffer. . . . .	523	parent_uow_id - Parent unit of work ID monitor element . . . . .	539
num_log_part_page_io - Number of Partial Log Page Writes . . . . .	523	partial_record - Partial Record monitor element	539
num_log_read_io - Number of Log Reads . . . . .	524	participant_no - Participant within Deadlock . . . . .	540
num_log_write_io - Number of Log Writes . . . . .	524	participant_no_holding_lk - Participant Holding a Lock on the Object Required by Application . . . . .	540
num_nodes_in_db2_instance - Number of Nodes in Partition . . . . .	524	partition_number - Partition Number . . . . .	541
num_remaps - Number of remaps monitor element	525	passthru_time - Pass-Through Time . . . . .	541
num_threshold_violations - Number of threshold violations monitor element . . . . .	525	passthru - Pass-Through . . . . .	541
num_transmissions - Number of Transmissions . . . . .	525	pipedsorts_accepted - Piped Sorts Accepted . . . . .	542
num_transmissions_group - Number of Transmissions Group . . . . .	526	pipedsorts_requested - Piped Sorts Requested . . . . .	542
number_in_bin - Number in bin monitor element	526	pkg_cache_inserts - Package Cache Inserts. . . . .	543
olap_func_overflows - OLAP Function Overflows monitor element . . . . .	527	pkg_cache_lookups - Package Cache Lookups . . . . .	543
open_cursors - Number of Open Cursors . . . . .	527	pkg_cache_num_overflows - Package Cache Overflows . . . . .	545
open_loc_curs - Open Local Cursors. . . . .	528	pkg_cache_size_top - Package cache high watermark . . . . .	545
open_loc_curs_blk - Open Local Cursors with Blocking . . . . .	528	pool_async_data_read_reqs - Buffer pool asynchronous read requests monitor element. . . . .	546
open_rem_curs - Open Remote Cursors . . . . .	528		
open_rem_curs_blk - Open Remote Cursors with Blocking . . . . .	529		



pool_async_data_reads - Buffer pool asynchronous data reads monitor element . . . . .	547	pool_write_time - Total buffer pool physical write time monitor element . . . . .	584
pool_async_data_writes - Buffer pool asynchronous data writes monitor element . . . . .	547	pool_xda_l_reads - Buffer pool XDA data logical reads monitor element . . . . .	585
pool_async_index_read_reqs - Buffer pool asynchronous index read requests monitor element.	548	pool_xda_p_reads - Buffer pool XDA data physical reads monitor element . . . . .	588
pool_async_index_reads - Buffer pool asynchronous index reads monitor element . . . . .	549	pool_xda_writes - Buffer pool XDA data writes monitor element . . . . .	589
pool_async_index_writes - Buffer pool asynchronous index writes monitor element . . . . .	550	post_shrthreshold_hash_joins - Post threshold hash joins . . . . .	591
pool_async_read_time - Buffer Pool Asynchronous Read Time . . . . .	551	post_shrthreshold_sorts - Post shared threshold sorts monitor element . . . . .	591
pool_async_write_time - Buffer Pool Asynchronous Write Time . . . . .	551	post_threshold_hash_joins - Hash Join Threshold	593
pool_async_xda_read_reqs - Buffer pool asynchronous XDA read requests monitor element . . . . .	552	post_threshold_olap_funcs - OLAP Function Threshold monitor element . . . . .	593
pool_async_xda_reads - Buffer pool asynchronous XDA data reads monitor element . . . . .	553	post_threshold_sorts - Post threshold sorts monitor element . . . . .	593
pool_async_xda_writes - Buffer pool asynchronous XDA data writes monitor element . . . . .	554	prefetch_wait_time - Time waited for prefetch monitor element . . . . .	595
pool_config_size - Configured Size of Memory Pool	554	prep_time - Preparation time monitor element . . . . .	595
pool_cur_size - Current Size of Memory Pool . . . . .	555	prep_time_best - Statement best preparation time monitor element . . . . .	596
pool_data_l_reads - Buffer pool data logical reads monitor element . . . . .	556	prep_time_worst - Statement worst preparation time monitor element . . . . .	596
pool_data_p_reads - Buffer pool data physical reads monitor element . . . . .	557	prev_uow_stop_time - Previous Unit of Work Completion Timestamp . . . . .	596
pool_data_writes - Buffer pool data writes monitor element . . . . .	559	priv_workspace_num_overflows - Private Workspace Overflows . . . . .	597
pool_drty_pg_steal_clns - Buffer pool victim page cleaners triggered monitor element . . . . .	561	priv_workspace_section_inserts - Private Workspace Section Inserts . . . . .	597
pool_drty_pg_thrsh_clns - Buffer pool threshold cleaners triggered monitor element . . . . .	562	priv_workspace_section_lookups - Private Workspace Section Lookups . . . . .	598
pool_id - Memory Pool Identifier . . . . .	563	priv_workspace_size_top - Maximum Private Workspace Size . . . . .	599
pool_index_l_reads - Buffer pool index logical reads monitor element . . . . .	564	product_name - Product Name . . . . .	599
pool_index_p_reads - Buffer pool index physical reads monitor element . . . . .	566	progress_completed_units - Completed Progress Work Units . . . . .	599
pool_index_writes - Buffer pool index writes monitor element . . . . .	567	progress_description - Progress Description . . . . .	600
pool_lsn_gap_clns - Buffer pool log space cleaners triggered monitor element . . . . .	569	progress_list_attr - Current Progress List Attributes	600
pool_no_victim_buffer - Buffer pool no victim buffers monitor element . . . . .	570	progress_list_cur_seq_num - Current Progress List Sequence Number . . . . .	601
pool_read_time - Total buffer pool physical read time monitor element . . . . .	571	progress_seq_num - Progress Sequence Number	601
pool_secondary_id - Memory Pool Secondary Identifier . . . . .	573	progress_start_time - Progress Start Time . . . . .	601
pool_temp_data_l_reads - Buffer pool temporary data logical reads monitor element . . . . .	573	progress_total_units - Total Progress Work Units	601
pool_temp_data_p_reads - Buffer pool temporary data physical reads monitor element . . . . .	575	progress_work_metric - Progress Work Metric . . . . .	602
pool_temp_index_l_reads - Buffer pool temporary index logical reads monitor element . . . . .	577	pseudo_deletes - Pseudo deletes monitor element	602
pool_temp_index_p_reads - Buffer pool temporary index physical reads monitor element . . . . .	578	monitor element . . . . .	602
pool_temp_xda_l_reads - Buffer pool temporary XDA data logical reads monitor element . . . . .	580	qp_query_id - Query patroller query ID monitor element . . . . .	603
pool_temp_xda_p_reads - Buffer pool temporary XDA data physical reads monitor element . . . . .	582	query_card_estimate - Query Number of Rows Estimate . . . . .	603
pool_watermark - Memory Pool Watermark . . . . .	583	query_cost_estimate - Query cost estimate monitor element . . . . .	604
		queue_assignments_total - Queue assignments total monitor element . . . . .	604
		queue_size_top - Queue size top monitor element	605
		queue_time_total - Queue time total monitor element . . . . .	605
		quiescer_agent_id - Quiescer Agent Identification	605

quiescer_auth_id - Quiescer User Authorization Identification . . . . .	605	rolled_back_appl_id - Rolled Back Application . . .	620
quiescer_obj_id - Quiescer Object Identification . .	606	rolled_back_participant_no - Rolled back application participant monitor element . . . .	620
quiescer_state - Quiescer State . . . . .	606	rolled_back_sequence_no - Rolled Back Sequence Number . . . . .	621
quiescer_ts_id - Quiescer Table Space Identification	606	root_node_splits - Root node splits monitor element . . . . .	621
range_adjustment - Range Adjustment . . . . .	607	routine_id - Routine ID monitor element . . . . .	621
range_container_id - Range Container . . . . .	607	rows_deleted - Rows deleted monitor element . . .	621
range_end_stripe - End Stripe . . . . .	607	rows_fetched - Rows fetched monitor element . . .	622
range_max_extent - Maximum Extent in Range . . .	607	rows_inserted - Rows inserted monitor element	622
range_max_page_number - Maximum Page in Range . . . . .	608	rows_modified - Rows modified monitor element	623
range_num_containers - Number of Containers in Range . . . . .	608	rows_read - Rows read monitor element . . . . .	624
range_number - Range Number . . . . .	608	rows_returned - Rows returned monitor element	626
range_offset - Range Offset . . . . .	608	rows_returned_top - Actual rows returned top monitor element . . . . .	627
range_start_stripe - Start Stripe . . . . .	608	rows_selected - Rows Selected . . . . .	628
range_stripe_set_number - Stripe Set Number . . .	609	rows_updated - Rows updated monitor element	628
reclaimable_space_enabled - Reclaimable space enabled indicator monitor element . . . . .	609	rows_written - Rows Written . . . . .	629
rej_curs_blk - Rejected Block Cursor Requests . .	609	rqsts_completed_total - Total requests completed monitor element . . . . .	630
rem_cons_in - Remote Connections To Database Manager . . . . .	610	sc_work_action_set_id - Service class work action set ID monitor element . . . . .	630
rem_cons_in_exec - Remote Connections Executing in the Database Manager . . . . .	610	sc_work_class_id - Service class work class ID monitor element . . . . .	631
remote_lock_time - Remote Lock Time . . . . .	611	sec_log_used_top - Maximum Secondary Log Space Used . . . . .	631
remote_locks - Remote Locks . . . . .	611	sec_logs_allocated - Secondary Logs Allocated Currently . . . . .	632
reorg_completion - Reorganization Completion Flag	611	section_env - Section environment monitor element	632
reorg_current_counter - Reorganize Progress . . .	612	section_number - Section number monitor element	633
reorg_end - Table Reorganize End Time . . . . .	612	section_type - Section type indicator monitor element . . . . .	633
reorg_index_id - Index Used to Reorganize the Table . . . . .	612	select_sql_stmts - Select SQL Statements Executed	634
reorg_long_tbspc_id - Table Space Where Long Objects are Reorganized monitor element . . . . .	612	select_time - Query Response Time . . . . .	634
reorg_max_counter - Total Amount of Reorganization . . . . .	613	sequence_no - Sequence number monitor element	635
reorg_max_phase - Maximum Reorganize Phase	613	sequence_no_holding_lk - Sequence Number Holding Lock . . . . .	636
reorg_phase - Table reorganization phase monitor element . . . . .	613	server_db2_type - Database Manager Type at Monitored (Server) Node . . . . .	636
reorg_phase_start - Reorganize Phase Start Time	614	server_instance_name - Server Instance Name . . .	637
reorg_rows_compressed - Rows Compressed . . . .	614	server_platform - Server Operating System . . . .	637
reorg_rows_rejected_for_compression - Rows Rejected for Compression . . . . .	614	server_prdid - Server Product/Version ID . . . .	637
reorg_start - Table Reorganize Start Time . . . .	615	server_version - Server Version . . . . .	638
reorg_status - Table Reorganize Status . . . . .	615	service_class_id - Service class ID monitor element	639
reorg_tbspc_id - Table Space Where Table or Data partition is Reorganized . . . . .	615	service_level - Service Level . . . . .	639
reorg_type - Table Reorganize Attributes . . . .	616	service_subclass_name - Service subclass name monitor element . . . . .	640
reorg_xml_regions_compressed - XML regions compressed monitor element . . . . .	616	service_superclass_name - Service superclass name monitor element . . . . .	640
reorg_xml_regions_rejected_for_compression - XML regions rejected for compression monitor element . . . . .	617	session_auth_id - Session authorization ID monitor element . . . . .	641
request_exec_time_avg - Request execution time average monitor element . . . . .	617	shr_workspace_num_overflows - Shared Workspace Overflows . . . . .	642
rf_log_num - Log Being Rolled Forward . . . . .	617	shr_workspace_section_inserts - Shared Workspace Section Inserts . . . . .	642
rf_status - Log Phase . . . . .	618	shr_workspace_section_lookups - Shared Workspace Section Lookups . . . . .	643
rf_timestamp - Rollforward Timestamp . . . . .	618	shr_workspace_size_top - Maximum Shared Workspace Size . . . . .	643
rf_type - Rollforward Type . . . . .	618		
rollback_sql_stmts - Rollback Statements Attempted . . . . .	619		
rolled_back_agent_id - Rolled Back Agent . . . .	620		

smallest_log_avail_node - Node with Least Available Log Space . . . . .	644	status_change_time - Application Status Change Time . . . . .	660
sort_heap_allocated - Total Sort Heap Allocated . . . . .	644	stmt_elapsed_time - Most Recent Statement Elapsed Time . . . . .	660
sort_heap_top - Sort private heap high watermark . . . . .	645	stmt_first_use_time - Statement first use time. . . . .	661
sort_overflows - Sort overflows monitor element . . . . .	645	stmt_history_id - Statement history identifier. . . . .	661
sort_shrheap_allocated - Sort Share Heap Currently Allocated. . . . .	647	inact_stmthist_sz - Statement history list size. . . . .	661
sort_shrheap_top - Sort share heap high watermark . . . . .	647	stmt_invocation_id - Statement invocation identifier monitor element . . . . .	662
source_service_class_id - Source service class ID monitor element . . . . .	648	stmt_isolation - Statement isolation . . . . .	662
sp_rows_selected - Rows Returned by Stored Procedures . . . . .	648	stmt_last_use_time - Statement last use time monitor element . . . . .	663
sql_chains - Number of SQL Chains Attempted . . . . .	649	stmt_lock_timeout - Statement lock timeout monitor element . . . . .	663
sql_req_id - Request Identifier for SQL Statement . . . . .	649	stmt_nest_level - Statement nesting level monitor element . . . . .	664
sql_reqs_since_commit - SQL Requests Since Last Commit . . . . .	649	stmt_node_number - Statement Node . . . . .	664
sql_stmts - Number of SQL Statements Attempted . . . . .	650	stmt_operation/operation - Statement operation monitor element . . . . .	664
sqlca - SQL Communications Area (SQLCA) . . . . .	650	stmt_pkgcache_id - Statement package cache identifier . . . . .	666
sqlrowsread_threshold_id - SQL rows read threshold ID monitor element . . . . .	651	stmt_query_id - Statement query identifier monitor element . . . . .	666
sqlrowsread_threshold_value - SQL rows read threshold value monitor element . . . . .	651	stmt_sorts - Statement Sorts . . . . .	667
sqlrowsread_threshold_violated - SQL rows read threshold violated monitor element . . . . .	651	stmt_source_id - Statement source identifier . . . . .	667
sqlrowsreadinsc_threshold_id - SQL rows read in service class threshold ID monitor element . . . . .	652	stmt_start - Statement Operation Start Timestamp . . . . .	668
sqlrowsreadinsc_threshold_value - SQL rows read in service class threshold value monitor element . . . . .	652	stmt_stop - Statement Operation Stop Timestamp . . . . .	668
sqlrowsreadinsc_threshold_violated - SQL rows read in service class threshold violated monitor element . . . . .	652	stmt_sys_cpu_time - System CPU Time used by Statement . . . . .	669
sqlrowsreturned_threshold_id - SQL rows read returned threshold ID monitor element. . . . .	653	stmt_text - SQL statement text monitor element . . . . .	669
sqlrowsreturned_threshold_value - SQL rows read returned threshold value monitor element. . . . .	653	stmt_type - Statement type monitor element . . . . .	670
sqlrowsreturned_threshold_violated - SQL rows read returned threshold violated monitor element . . . . .	653	stmt_usr_cpu_time - User CPU Time used by Statement . . . . .	671
sqltempespace_threshold_id - SQL temporary space threshold ID monitor element . . . . .	654	stmt_value_data - Value data . . . . .	672
sqltempespace_threshold_value - SQL temporary space threshold value monitor element. . . . .	654	stmt_value_index - Value index . . . . .	672
sqltempespace_threshold_violated - SQL temporary space threshold violated monitor element . . . . .	654	stmt_value_isnull - Value has null value monitor element . . . . .	672
ss_exec_time - Subsection Execution Elapsed Time . . . . .	655	stmt_value_isreopt - Variable used for statement reoptimization monitor element . . . . .	673
ss_node_number - Subsection Node Number. . . . .	655	stmt_value_type - Value type monitor element . . . . .	673
ss_number - Subsection Number . . . . .	655	sto_path_free_sz - Automatic Storage Path Free Space . . . . .	674
ss_status - Subsection Status . . . . .	656	stop_time - Event Stop Time . . . . .	674
ss_sys_cpu_time - System CPU Time used by Subsection . . . . .	656	stored_proc_time - Stored Procedure Time. . . . .	675
ss_usr_cpu_time - User CPU Time used by Subsection . . . . .	656	stored_procs - Stored Procedures . . . . .	675
start_time - Event Start Time . . . . .	657	sync_runstats - Total number of synchronous RUNSTATS activities monitor element . . . . .	675
static_sql_stmts - Static SQL Statements Attempted . . . . .	657	sync_runstats_time - Total time spent on synchronous RUNSTATS activities monitor element . . . . .	676
statistics_timestamp - Statistics timestamp monitor element . . . . .	658	system_cpu_time - System CPU Time . . . . .	677
stats_cache_size - Size of statistics cache monitor element . . . . .	658	tab_file_id - Table file ID monitor element. . . . .	677
stats_fabricate_time - Total time spent on statistics fabrication activities monitor element . . . . .	659	tab_type - Table type monitor element . . . . .	677
stats_fabrications - Total number of statistics fabrications monitor elements . . . . .	660	table_file_id - Table file ID monitor element . . . . .	678
		table_name - Table name monitor element. . . . .	678
		table_scans - Table scans monitor element . . . . .	679
		table_schema - Table schema name monitor element . . . . .	679
		table_type - Table type monitor element . . . . .	681
		tablespace_auto_resize_enabled - Table space automatic resizing enabled monitor element . . . . .	681



tablespace_content_type - Table space content type monitor element . . . . .	682	tablespace_state_change_ts_id - State Change Table Space Identification . . . . .	695
tablespace_cur_pool_id - Buffer pool currently being used monitor element . . . . .	682	tablespace_total_pages - Total pages in table space monitor element . . . . .	696
tablespace_current_size - Current table space size monitor element . . . . .	683	tablespace_type - Table space type monitor element . . . . .	696
tablespace_extent_size - Table space extent size monitor element . . . . .	683	tablespace_usable_pages - Usable pages in table space monitor element . . . . .	697
tablespace_free_pages - Free pages in table space monitor element . . . . .	683	tablespace_used_pages - Used pages in table space monitor element . . . . .	697
tablespace_id - Table space identification monitor element . . . . .	684	tablespace_using_auto_storage - Table space enabled for automatic storage monitor element . . . . .	698
tablespace_increase_size - Increase size in bytes monitor element . . . . .	684	tbbsp_max_page_top - Maximum table space page high watermark monitor element . . . . .	698
tablespace_increase_size_percent - Increase size by percent monitor element . . . . .	685	tcpip_rcv_volume - TCP/IP received volume monitor element . . . . .	698
tablespace_initial_size - Initial table space size . . . . .	685	tcpip_rcv_wait_time - TCP/IP received wait time monitor element . . . . .	699
tablespace_last_resize_failed - Last resize attempt failed . . . . .	685	tcpip_rcvs_total - TCP/IP receives total monitor element . . . . .	700
tablespace_last_resize_time - Time of last successful resize . . . . .	686	tcpip_send_volume - TCP/IP send volume monitor element . . . . .	701
tablespace_max_size - Maximum table space size . . . . .	686	tcpip_send_wait_time - TCP/IP send wait time monitor element . . . . .	702
tablespace_min_recovery_time - Minimum Recovery Time For Rollforward . . . . .	686	tcpip_sends_total - TCP/IP sends total monitor element . . . . .	702
tablespace_name - Table space name monitor element . . . . .	687	temp_tablespace_top - Temporary table space top monitor element . . . . .	703
tablespace_next_pool_id - Buffer pool that will be used at next startup monitor element . . . . .	688	territory_code - Database Territory Code . . . . .	704
tablespace_num_containers - Number of Containers in Table Space . . . . .	688	threshold_action - Threshold action monitor element . . . . .	704
tablespace_num_quiescers - Number of Quiescers . . . . .	688	threshold_domain - Threshold domain monitor element . . . . .	705
tablespace_num_ranges - Number of Ranges in the Table Space Map . . . . .	689	threshold_maxvalue - Threshold maximum value monitor element . . . . .	705
tablespace_page_size - Table space page size monitor element . . . . .	689	threshold_name - Threshold name monitor element . . . . .	705
tablespace_page_top - Table space high watermark monitor element . . . . .	689	threshold_predicate - Threshold predicate monitor element . . . . .	706
tablespace_paths_dropped - Table space using dropped path monitor element . . . . .	689	threshold_queuesize - Threshold queue size monitor element . . . . .	706
tablespace_pending_free_pages - Pending free pages in table space monitor element . . . . .	690	thresholdid - Threshold ID monitor element . . . . .	706
tablespace_prefetch_size - Table space prefetch size monitor element . . . . .	690	time_completed - Time completed monitor element . . . . .	707
tablespace_rebalancer_extents_processed - Number of Extents the Rebalancer has Processed . . . . .	691	time_created - Time created monitor element . . . . .	707
tablespace_rebalancer_extents_remaining - Total Number of Extents to be Processed by the Rebalancer . . . . .	691	time_of_violation - Time of violation monitor element . . . . .	707
tablespace_rebalancer_last_extent_moved - Last Extent Moved by the Rebalancer . . . . .	692	time_stamp - Snapshot Time . . . . .	708
tablespace_rebalancer_mode - Rebalancer mode monitor element . . . . .	692	time_started - Time started monitor element . . . . .	708
tablespace_rebalancer_priority - Current Rebalancer Priority . . . . .	693	time_zone_disp - Time Zone Displacement . . . . .	708
tablespace_rebalancer_restart_time - Rebalancer Restart Time. . . . .	693	top - Histogram bin top monitor element . . . . .	708
tablespace_rebalancer_start_time - Rebalancer Start Time . . . . .	694	tot_log_used_top - Maximum Total Log Space Used . . . . .	709
tablespace_state - Table space state monitor element . . . . .	694	total_act_time - Total activity time monitor element . . . . .	709
tablespace_state_change_object_id - State Change Object Identification . . . . .	695	total_act_wait_time - Total activity wait time monitor element . . . . .	709
		total_app_rqst_time - Total application request time monitor element . . . . .	710
		total_buffers_rcvd - Total FCM Buffers Received . . . . .	711
		total_buffers_sent - Total FCM Buffers Sent . . . . .	711
		total_cons - Connects Since Database Activation . . . . .	711
		total_cpu_time - Total CPU time monitor element . . . . .	712
		total_exec_time - Elapsed Statement Execution Time . . . . .	713

total_move_time - Total extent move time monitor element . . . . .	713
total_hash_joins - Total Hash Joins . . . . .	713
total_hash_loops - Total Hash Loops. . . . .	714
total_log_available - Total Log Available . . . . .	714
total_log_used - Total Log Space Used . . . . .	715
total_olap_funcs - Total OLAP Functions monitor element . . . . .	715
total_rqst_mapped_in - Total request mapped-in monitor element . . . . .	716
total_rqst_mapped_out - Total request mapped-out monitor element . . . . .	716
total_rqst_time - Total request time monitor element . . . . .	716
total_sec_cons - Secondary Connections . . . . .	717
total_section_sorts - Total section sorts monitor element . . . . .	717
total_section_sort_proc_time - Total section sort processing time monitor element . . . . .	718
total_section_sort_time - Total section sort time monitor element . . . . .	719
total_sort_time - Total sort time monitor element	720
total_sorts - Total sorts monitor element . . . . .	721
total_sys_cpu_time - Total system CPU time for a statement monitor element . . . . .	723
total_sorts - Total sorts monitor element . . . . .	723
total_usr_cpu_time - Total user CPU time for a statement monitor element . . . . .	725
total_wait_time - Total wait time monitor element	725
tpmon_acc_str - TP monitor client accounting string monitor element . . . . .	726
tpmon_client_app - TP monitor client application name monitor element . . . . .	726
tpmon_client_userid - TP monitor client user ID monitor element . . . . .	727
tpmon_client_wkstn - TP monitor client workstation name monitor element . . . . .	727
tq_cur_send_spills - Current number of table queue buffers overflowed monitor element. . . . .	728
tq_id_waiting_on - Waited on node on a table queue . . . . .	728
tq_max_send_spills - Maximum number of table queue buffers overflows. . . . .	729
tq_node_waited_for - Waited for node on a table queue . . . . .	729
tq_rows_read - Number of Rows Read from table queues . . . . .	729
tq_rows_written - Number of rows written to table queues . . . . .	730
tq_tot_send_spills - Total number of table queue buffers overflowed monitor element. . . . .	730
tq_wait_for_any - Waiting for any node to send on a table queue . . . . .	731
ts_name - Table space being rolled forward monitor element . . . . .	732
uid_sql_stmts - Update/Insert/Delete SQL Statements Executed . . . . .	732
unread_prefetch_pages - Unread prefetch pages monitor element . . . . .	733
uow_comp_status - Unit of Work Completion Status . . . . .	733

uow_elapsed_time - Most Recent Unit of Work Elapsed Time . . . . .	734
uow_id - Unit of work ID monitor element . . . . .	734
uow_lock_wait_time - Total time unit of work waited on locks monitor element . . . . .	735
uow_log_space_used - Unit of Work Log Space Used . . . . .	735
uow_start_time - Unit of Work Start Timestamp	736
uow_status - Unit of Work Status. . . . .	737
uow_stop_time - Unit of work stop timestamp monitor element . . . . .	737
uow_total_time_top - UOW total time top monitor element . . . . .	738
update_sql_stmts - Updates . . . . .	738
update_time - Update Response Time . . . . .	738
user_cpu_time - User CPU Time . . . . .	739
utility_dbname - Database Operated on by Utility	739
utility_description - Utility Description . . . . .	739
utility_id - Utility ID . . . . .	740
utility_invoker_type - Utility Invoker Type . . . . .	740
utility_priority - Utility Priority . . . . .	740
utility_start_time - Utility Start Time . . . . .	740
utility_state - Utility State . . . . .	741
valid - Section validity indicator monitor element	741
utility_type - Utility Type . . . . .	741
vectored_ios - Number of vectored I/O requests monitor element . . . . .	742
version - Version of Monitor Data . . . . .	742
wlm_queue_assignments_total - Workload manager total queue assignments monitor element . . . . .	742
wlm_queue_time_total - Workload manager total queue time monitor element . . . . .	743
wlo_completed_total - Workload occurrences completed total monitor element . . . . .	744
work_action_set_id - Work action set ID monitor element . . . . .	744
work_action_set_name - Work action set name monitor element . . . . .	745
work_class_id - Work class ID monitor element . . . . .	745
work_class_name - Work class name monitor element . . . . .	745
workload_id - Workload ID monitor element . . . . .	745
workload_name - Workload name monitor element	746
workload_occurrence_id - Workload occurrence identifier monitor element . . . . .	747
workload_occurrence_state - Workload occurrence state monitor element . . . . .	748
x_lock_escals - Exclusive Lock Escalations. . . . .	748
xda_object_pages - XDA Object Pages . . . . .	749
xid - Transaction ID . . . . .	750
xquery_stmts - XQuery Statements Attempted . . . . .	750

---

## Part 3. Appendixes . . . . . 751

### Appendix A. Overview of the DB2 technical information . . . . . 753

DB2 technical library in hardcopy or PDF format	753
Ordering printed DB2 books . . . . .	756
Displaying SQL state help from the command line processor. . . . .	757

Accessing different versions of the DB2 Information Center . . . . .	757
Displaying topics in your preferred language in the DB2 Information Center . . . . .	757
Updating the DB2 Information Center installed on your computer or intranet server . . . . .	758
Manually updating the DB2 Information Center installed on your computer or intranet server . . . . .	759

DB2 tutorials . . . . .	761
DB2 troubleshooting information . . . . .	761
Terms and Conditions . . . . .	762

**Appendix B. Notices . . . . . 763**

**Index . . . . . 767**

---

## About this book

The *System Monitor Guide and Reference* describes how to collect different kinds of information about your database and the database manager.

It also explains how you can use the information you collected to understand database activity, improve performance, and determine the cause of problems.



---

## **Part 1. Monitoring interfaces**





---

## Chapter 1. Database monitoring

Database monitoring is a vital activity for the maintenance of the performance and health of your database management system. To facilitate monitoring, DB2® collects information from the database manager, its databases, and any connected applications. With this information you can do the following, and more:

- Forecast hardware requirements based on database usage patterns.
- Analyze the performance of individual applications or SQL queries.
- Track the usage of indexes and tables.
- Pinpoint the cause of poor system performance.
- Assess the impact of optimization activities (for instance, altering database manager configuration parameters, adding indexes, or modifying SQL queries).



---

## Chapter 2. Monitor table functions overview

Starting with DB2 Version 9.7, you can access monitor data through a light-weight alternative to the traditional system monitor. Use monitor table functions to collect and view data for systems, activities, or data objects.

Data for monitored elements are continually accumulated in memory and available for querying. You can choose to receive data for a single object (for example, service class A or table TABLE1) or for all objects.

When using these table functions in a database partitioned environment, you can choose to receive data for a single partition or for all partitions. If you choose to receive data for all partitions, the table functions return one row for each partition. Using SQL, you can sum the values across partitions to obtain the value of a monitor element across partitions.

---

### Monitoring system information using table functions

The system monitoring perspective encompasses the complete volume of work and effort expended by the data server to process application requests. From this perspective, you can determine what the data server is doing as a whole as well as for particular subsets of application requests.

Monitor elements for this perspective, referred to as request monitor elements, cover the entire range of data server operations associated with processing requests.

Request monitor elements are continually accumulated and aggregated in memory so they are immediately available for querying. Request monitor elements are aggregated across requests at various levels of the workload management (WLM) object hierarchy: by unit of work, by workload, by service class. They are also aggregated by connection.

Use the following table functions for accessing current system monitoring information:

- MON\_GET\_SERVICE\_SUBCLASS and MON\_GET\_SERVICE\_SUBCLASS\_DETAILS
- MON\_GET\_WORKLOAD and MON\_GET\_WORKLOAD\_DETAILS
- MON\_GET\_CONNECTION and MON\_GET\_CONNECTION\_DETAILS
- MON\_GET\_UNIT\_OF\_WORK and MON\_GET\_UNIT\_OF\_WORK\_DETAILS

This set of table functions enables you to drill down or focus on request monitor elements at a particular level of aggregation. Table functions are provided in pairs: one for relational access to commonly used data and the other for XML access to the complete set of available monitor elements.

The system monitoring information is collected by these table functions by default for a new database. You can change default settings using one or both of the following settings:

- The database configuration parameter **mon\_req\_metrics** specifies the minimum level of collection in all service classes.

- The COLLECT REQUEST METRICS clause of the CREATE/ALTER SERVICE CLASS statement specifies the level of collection for a service superclass. Use this setting to increase the level of collection for a given service class over the minimum level of collection set for all service classes.

The possible values for each setting are the following:

**None** No request monitor elements are collected

**Base** All request monitor elements are collected

For example, to collect system monitoring information for only a subset of service classes, do the following:

1. Set the database configuration parameter **mon\_req\_metrics** to NONE.
2. For each desired service class, set the COLLECT REQUEST METRICS clause of the CREATE/ALTER SERVICE CLASS statement to BASE.

---

## Monitoring activities using table functions

The activity monitoring perspective focuses on the subset of data server processing related to executing activities. In the context of SQL statements, the term activity refers to the execution of the section for a SQL statement.

Monitor elements for this perspective, referred to as activity monitor elements, are a subset of the request monitor elements. Activity monitor elements measure aspects of work done for statement section execution. Activity monitoring includes other information such as SQL statement text for the activity.

For activities in progress, activity metrics are accumulated in memory. For activities that are SQL statements, activity metrics are also accumulated in the package cache. In the package cache activity metrics are aggregated over all executions of each SQL statement section.

Use the following table functions to access current data for activities:

### **MON\_GET\_ACTIVITY\_DETAILS**

Returns data about the individual activities in progress when the table function is called. Data is returned in XML format.

### **MON\_GET\_PKG\_CACHE\_STMT**

Returns data for individual SQL statement section aggregated over all executions of the section. Data is returned in a relational form.

Activity monitoring information is collected by default for a new database. You can change default settings using one or both of the following settings:

- The **mon\_act\_metrics** database configuration parameter specifies the minimum level of collection in all workloads.
- The COLLECT ACTIVITY METRICS clause of the CREATE/ALTER WORKLOAD statement specifies the level of collection for a given workload over the minimum level of collection set for all workloads.

The possible values for each setting are the following:

**None** No activity monitor elements are collected

**Base** All activity monitor elements are collected

For example, to collect activity monitor elements for only selected workloads, do the following:

1. Set the **mon\_act\_metrics** database configuration parameter to NONE.
2. Set the COLLECT ACTIVITY METRICS clause of the CREATE/ALTER WORKLOAD statement to BASE. By default, the values for other workloads is NONE.

---

## Monitoring data objects using table functions

The data object monitoring perspective provides information about operations performed on data objects, that is tables, indexes, buffer pools, table spaces, and containers.

A different set of monitor elements is available for each object type. Monitor elements for a data object are incremented each time a request involves processing that object. For example, when processing a request that involves reading rows from a particular table, the metric for rows read is incremented for that table.

Use the following table functions to access current details for data objects:

- MON\_GET\_BUFFERPOOL
- MON\_GET\_TABLESPACE
- MON\_GET\_CONTAINER
- MON\_GET\_TABLE
- MON\_GET\_INDEX

These table functions return data in a relational form.

You cannot access historical data for data objects.

Data object monitor elements are collected by default for new databases. You can use the **mon\_obj\_metrics** database configuration parameter to reduce the amount of data collected by the table functions.

The possible values for this configuration parameter are the following:

- None** No data object monitor elements are collected
- Base** All data object monitor elements are collected

Regardless of the what you set the **mon\_obj\_metrics** parameter to, data is always collected for monitor elements reported by the following table functions:

- MON\_GET\_TABLE
- MON\_GET\_INDEX

To stop collecting data object monitor elements reported by the following table functions, set the **mon\_obj\_metrics** configuration parameter to NONE.

- MON\_GET\_BUFFERPOOL
- MON\_GET\_TABLESPACE
- MON\_GET\_CONTAINER



---

## Chapter 3. Event monitors

Event monitors return information for the event types specified in the CREATE EVENT MONITOR statement. For each event type, monitoring information is collected at a certain point in time.

The following table lists available event types, when the monitoring data is collected, and the information available for each event type. The available event types in the first column correspond to the keywords used in the CREATE EVENT MONITOR statement, where the event type is defined.

In addition to the defined events where data occurs, you can use the FLUSH EVENT MONITOR SQL statement to generate events. The events generated by this method are written with the current database monitor values for all the monitor types (except for DEADLOCKS and DEADLOCKS WITH DETAILS) associated with the flushed event monitor.

When monitoring the execution of SQL procedures using statement event monitors:

- Data manipulation language (DML) statements, such as INSERT, SELECT, DELETE, and UPDATE, generate events.
- Procedural statements, such as variable assignments and control structures (for example, WHILE or IF), do not generate events in a deterministic fashion.

Table 1. Event Types

Event type	When data is collected	Available information
DEADLOCKS <sup>1</sup>	Detection of a deadlock	Applications involved, and locks in contention.
DEADLOCKS WITH DETAILS <sup>1</sup>	Detection of a deadlock	Comprehensive information regarding applications involved, including the identification of participating statements (and statement text) and a list of locks being held. Using a DEADLOCKS WITH DETAILS event monitor instead of a DEADLOCKS event monitor will incur a performance cost when deadlocks occur, due to the extra information that is collected.
DEADLOCKS WITH DETAILS HISTORY <sup>1</sup>	Detection of a deadlock	All information reported in a DEADLOCKS WITH DETAILS event monitor, along with the statement history for the current unit of work of each application owning a lock participating in a deadlock scenario for the database partition where that lock is held. Using a DEADLOCKS WITH DETAILS HISTORY event monitor will incur a minor performance cost when activated due to statement history tracking.
DEADLOCKS WITH DETAILS HISTORY VALUES <sup>1</sup>	Detection of a deadlock	All information reported in a deadlock with details and history, along with the values provided for any parameter markers at the time of execution of a statement. Using a DEADLOCKS WITH DETAILS HISTORY VALUES event monitor will incur a more significant performance cost when activated due to extra copying of data values.



Table 1. Event Types (continued)

Event type	When data is collected	Available information
STATEMENTS	End of SQL statement	Statement start or stop time, CPU used, text of dynamic SQL, SQLCA (return code of SQL statement), and other metrics such as fetch count. <b>Note:</b> Statement start or stop time is unavailable when the Timestamp switch is off.
	End of subsection	For partitioned databases: CPU consumed, execution time, table and table queue information.
TRANSACTIONS <sup>2</sup>	End of unit of work	UOW work start or stop time, previous UOW time, CPU consumed, locking and logging metrics. Transaction records are not generated if running with XA.
CONNECTIONS	End of connection	All application level counters.
DATABASE	Database deactivation	All database level counters.
BUFFERPOOLS	Database deactivation	Counters for buffer pool, prefetchers, page cleaners and direct I/O for each buffer pool.
TABLESPACES	Database deactivation	Counters for buffer pool, prefetchers, page cleaners and direct I/O for each table space.
TABLES	Database deactivation	Rows read or written for each table.
Activities	Completion of an activity that executed in a service class, workload or work class that had its COLLECT ACTIVITY DATA option turned on. Data is also collected for the targeted activity at the instant the WLM_CAPTURE_ACTIVITY_IN_PROGRESS stored procedure is executed.  Data is also collected if the activity violates a threshold that has the COLLECT ACTIVITY DATA option enabled.	Activity level data. If WITH DETAILS was specified as part of COLLECT ACTIVITY DATA, this will include statement and compilation environment information for those activities that have it. If AND VALUES was also specified, this will also include input data values for those activities that have it.
Statistics	Every <i>period</i> minutes, where <i>period</i> is the length of time over which statistics are gathered. This period is defined in the WLM_COLLECT_INT database configuration parameter.  Data is also collected when the WLM_COLLECT_STATS stored procedure is called.	Statistics computed from the activities that executed within each service class, workload, or work class that exists on the system.
Threshold violations	Upon detection of a threshold violation.	Threshold violation information.
Locking	Upon detection of any of the following event types, depending on configuration settings : lock timeout, deadlock, lock wait beyond a specified duration.	Lock event records.
Unit of work	Upon completion of a unit of work	Unit of work event records. Option to include request metrics in the record.

<sup>1</sup> This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

<sup>2</sup> This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR UNIT OF WORK statement to monitor transaction events.

**Note:** A detailed deadlock event monitor is created for each newly created database. This event monitor, named DB2DETAILDEADLOCK, starts when the database is activated and will write to files in the database directory. You can avoid

the overhead this event monitor incurs by dropping it. The DB2DETAILDEADLOCK event monitor is deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

---

## Event monitors that write to an unformatted event table

DB2 9.7 introduces some event monitors with a type of target, the unformatted event table. This type of event monitor provides better performance, new CREATE EVENT MONITOR statement options, and the new interfaces to access data for analysis.

Characteristics of unformatted event table event monitors affect how you can perform the following tasks:

- Creating the event monitor and configuring data collection
- Managing event monitor operations
- Accessing event data captured by the event monitor

Typically, you can achieve all your monitoring needs by creating a single event monitor per database for a given type of event (for example, lock events). You can alter settings to increase or decrease the amount of data that you can collect with the monitor to address changing monitoring needs. This contrasts with some older event monitors where a more common practice is to create a multiple event monitors, each geared to collect a particular monitoring need.

### Creating the unformatted event table associated with an event monitor

One aspect of creating an event monitor is specifying where to write the data that the monitor collects. This type of event monitor always writes data in binary format to an unformatted event table. The unformatted event table is a target type introduced in DB2 9.7. An unformatted event table is created implicitly each time you create an event monitor. The CREATE EVENT statement for this type of event monitor includes the clause WRITE TO UNFORMATTED EVENT TABLE.

The CREATE EVENT MONITOR statement includes the following options for configuring the unformatted event table:

- table name - By default, the unformatted event table is named based on the event monitor name.
- tablespace name - By default, the unformatted table is created in the table space IBMDEFAULTGROUP over which the user has USE privilege if it exists. However, the recommended practice is to define a tablespace optimized for your event monitors, as described below.
- PCTDEACTIVATE - The default value is 100, which means that the event monitor deactivates when the table space becomes full.

The following considerations about the table space for the unformatted event table must be taken:

- Create a table space for your event monitor unformatted event tables that is configured for performance. Use the following clauses with the CREATE TABLESPACE statement:
  - Specify a page size (PAGESIZE) as large as possible, up to and including 32KB.
  - Specify the NO FILE CACHING SYSTEM option.

- In a partitioned database environment, consider on which partitions the table space exists. If a table space for a target unformatted event table does not exist on some database partition, data for that target unformatted event table is ignored. This behavior allows users to choose a subset of database partitions for monitoring to be chosen, by creating a table space that exists only on certain database partitions.

Other useful information about unformatted event tables includes the following:

- The SYSCAT.EVENTTABLES catalog view lists event monitors, their associated unformatted table, and other details.
- The columns of the unformatted event table are described in a topic listed in the related links.

## Configuring data collection for an event monitor

Setting up an event monitor involves specifying what data to collect. Aspects include which subset of the system workload to monitor, what type of events to collect, how much detail to collect for each event, and enabling/disabling data capture (turning the data capture on and off). Considerations for configuring data collection are the following:

- With this type of event monitor, you configure data collection primarily by setting properties of individual workload definitions using the CREATE/ALTER WORKLOAD statement. That is, you can specify different data collection settings for different workloads. The CREATE/ALTER WORKLOAD statement includes clauses specific to particular types of event monitor.
- By default, this type of event monitor is automatically activated. You can specify that the event monitor be activated manually by specifying the MANUALSTART keyword in the CREATE EVENT statement. You can then control the event monitor with the SET EVENT MONITOR STATE statement.
- As mentioned in another context, in a partitioned database environment you can choose a subset of database partitions to monitor with your event monitor. When you create the event monitor, specify a table space for the unformatted event table that resides only on those partitions you want to monitor. If the unformatted event table does not exist on a given database partition, the event monitor will not collect data for that partition.
- Data collection for this type of event monitor is not affected by system monitor switch settings set using the UPDATE MONITOR SWITCHES statement nor is event capture is turned on and off using the SET EVENT MONITOR statement.

## Managing event monitor operations

The following points provide guidance for managing ongoing operation of an event monitor:

- At any time, you can change your specification of what data to collect by using the ALTER WORKLOAD statement.
- If you specified the MANUALSTART option in the CREATE EVENT statement, you can start and stop data collection using the SET EVENT MONITOR STATE statement.
- Unformatted event tables must be manually pruned.
- If an unformatted event table reaches the maximum space allotted, the event monitor will deactivate.
- If an event monitor is no longer needed, use the DROP statement to drop an event monitor. Issuing the DROP statement does not drop the unformatted event

table that is associated with the event monitor. The associated unformatted event table must be manually dropped after the event monitor is dropped. If you don't drop the unformatted event table, you will encounter difficulties if you subsequently try to create another event monitor whose unformatted event table has the same name as an existing one.

## Accessing event data captured by an event monitor

This type of event monitor writes data in a binary format to an unformatted event table. You can access this data using the `db2evmonfmt` command or routines provided for this purpose.

With the `db2evmonfmt` command you can:

- select events of interest based on the following attributes: event ID, event type, time period, application, workload, or service class.
- choose whether to receive the output in the form of a text report or a formatted XML document.
- completely control the output format by creating your own XSLT style sheets instead of using the ones provided with `db2evmonfmt`.

You can also extract data from an unformatted event table using the following routines:

- `EVMON_FORMAT_UE_TO_XML` - extracts data from an unformatted event table into an XML document.
- `EVMON_FORMAT_UE_TO_TABLES` - extracts data from an unformatted event table into a set of relational tables.

With these routines, you can use a `SELECT` statement to specify the exact rows from the unformatted event table that you want to extract.

## Unformatted event table column definitions

An unformatted event table is created when you issue a `CREATE EVENT` statement that includes the clause `WRITE TO UNFORMATTED EVENT TABLE`. The column definitions are useful when you want to extract data to analyze or prune a table of unneeded data.

The column definitions for the unformatted event table are useful when you want to extract data from an unformatted event table using one of the following routines:

- `EVMON_FORMAT_UE_TO_XML` - extracts data from an unformatted event table into an XML document.
- `EVMON_FORMAT_UE_TO_TABLES` - extracts data from an unformatted event table into a set of relational tables.

The call to these routines accepts a `SELECT` statement that specifies the rows that you want to extract. Use the unformatted event table column definitions to assist with composing your `SELECT` statement.

There is no automatic purging of the event data written to an unformatted event table. You must manually purge data from the table. The column definitions for the unformatted event table are useful when you want to purge a targeted set of records. Another option is to remove all the table rows using the `TRUNCATE TABLE` command.

As part of the CREATE EVENT MONITOR statement, you can specify what to name the associated unformatted event table. If not specified, the name defaults to the same name as the event monitor. The SYSCAT.EVENTTABLES catalog view lists event monitors, their associated unformatted table, and other details.

The table below describes the columns in the unformatted event table. The key column is the event\_data column. The other columns represent identifiers that you can use to locate events of interest. For further attributes of table columns, issue a DESCRIBE statement.

*Table 2. Unformatted event table column definitions*

Column name	Column data type	Column description
appl_id	VARCHAR	The identifier of the application within which the event occurred. A NULL value indicates that the application ID was not available.
appl_name	VARCHAR	The name of the application within which the event occurred. A NULL value indicates that the application name was not available.
client_acctng	VARCHAR	The current value of the CLIENT_ACCTNG special register for this event. A NULL value indicates that the client accounting was not available.
client_applname	VARCHAR	The current value of the CLIENT_APPLNAME special register for this event. A NULL value indicates that the client application name was not available.
client_userid	VARCHAR	The current value of the CLIENT_USERID special register for this event. A NULL value indicates that the client user ID was not available.
client_wrkstnname	VARCHAR	The current value of the CLIENT_WRKSTNNAME special register for this event. A NULL value indicates that the client workstation name was not available.

Table 2. Unformatted event table column definitions (continued)

Column name	Column data type	Column description
event_correlation_id	BIT DATA	<p>An optional event correlation ID. A NULL value indicates that the event correlation ID was not available.</p> <p>The value is based on the event monitor type:</p> <ul style="list-style-type: none"> <li>• LOCKING - Reserved for future use</li> <li>• UOW- Reserved for future use</li> </ul>
event_data	BLOB	The entire event record data for an event captured by the event monitor, stored in its original binary form.
event_id	INTEGER	<p>For locking event monitor records, an event identifier that is unique across the database. The ID is recycled at database activation time. Uniqueness is guaranteed by the combination of <b>event_timestamp, event_id, member, and event_type</b>.</p> <p>For UOW event monitor records, an alias of the UOW ID that is unique per connection. Uniqueness is guaranteed by the combination of <b>event_timestamp, event_id, member, event_type and appl_id</b>.</p>
event_timestamp	TIMESTAMP	The timestamp when the event was generated by the event monitor. All child records will share the same timestamp as the parent record.
event_type	VARCHAR	The event type that occurred at the member of detection.
member	SMALLINT	The member where the event occurred.
partitioning_key	INTEGER	The partitioning key for the table, so that insert operations are performed locally on the database partition where the event monitor is running.
record_seq_num	INTEGER	The sequence number of the record that is stored within the event_data column.

Table 2. Unformatted event table column definitions (continued)

Column name	Column data type	Column description
record_type	INTEGER	The type of record that is stored within the event_data column.
service_subclass_name	VARCHAR	The name of the service subclass within which the event occurred. A NULL value indicates that the service subclass name was not available.
service_superclass_name	VARCHAR	The name of the service superclass within which the event occurred. A NULL value indicates that the service superclass name was not available.
workload_name	VARCHAR	The name of the workload within which the event occurred. A NULL value indicates that the workload name was not available.

## db2evmonfmt tool for reading event monitor data

The Java™-based, generic XML parser tool, db2evmonfmt, produces a readable flat-text output (text version) or a formatted XML output from the data generated by an event monitor that uses the unformatted event table. Based on the parameters that you specify, the db2evmonfmt tool determines how to parse the event monitor data and the type of output to create.

The db2evmonfmt tool is provided as Java source code. You must setup and compile this tool, before you can use it, by performing the following steps:

1. Locate the source code in the `sql1lib/samples/java/jdbc` directory
2. Follow the instructions embedded in the Java source file to setup and compile the tool

You can modify the source code to change the output to your liking.

The tool uses XSLT style sheets to transform the event data into formatted text. You do not need to understand these style sheets. The tool will automatically load the correct style sheet, based on the event monitor type, and transform the event data. Each event monitor will provide default style sheets within the `sql1lib/samples/xml/data` directory. The tool will also provide the following filtering options:

- Event ID
- Event timestamp
- Event type
- Workload name
- Service class name
- Application name

## Tool syntax

```
▶▶ java db2evmonfmt [connect XML file] [filter options]
```

### connect:

```
| --d db_name --ue table_name [ -u user_id -p password ]
```

### XML file:

```
| -f xml_filename
```

### filter options:

```
| [ -fxml ] [ -ftext ] [ -ss stylesheet_name ] [ -id event_id ]  
▶▶ [ -type event_type ] [ -hours num_hours ] [ -w workload_name ]  
▶▶ [ -a appl_name ] [ -s srvc_subclass_name ]
```

## Tool parameters

### java

To run the db2evmonfmt Java-based tool successfully, the java keyword must precede the tool name. The proper Java version to successfully run this tool is installed from the `sql11ib/java/jdk64` directory during the DB2 product installation.

#### -d *db\_name*

Specifies the database name to which a connection is made

#### -ue *table\_name*

Specifies the name of the unformatted event table

#### -u *user\_id*

Specifies the user ID

#### -p *password*

Specifies the password

#### -f *xml\_filename*

Specifies the name of the input XML file to format

#### -fxml

Produces a formatted XML document (pipe to stdout)

#### -ftext

Formats an XML document to a text document (pipe to stdout)

#### -ss *stylesheet\_name*

Specifies the XSLT style sheet to use to transform the XML document



- id** *event\_id*  
Displays all events matching the specified event ID
- type** *event\_type*  
Displays all events matching the specified event type
- hours** *num\_hours*  
Displays all events that have occurred within the specified last number of hours
- w** *workload\_name*  
Displays all events that are part of the specified workload
- a** *appl\_name*  
Displays all events that are part of the specified application
- s** *srcv\_subclass\_name*  
Displays all events that are part of the specified service subclass

## XSLT style sheets

The DB2 database manager provides default XSLT style sheets (see Table 1) which can be found in the `sql1lib/samples/java/jdbc` directory. You can change these style sheets to produce the desired output.

*Table 3. Default XSLT style sheets for event monitors*

Event monitor	Default XSLT style sheet
Locking	DB2EvmonLocking.xsl
Unit of work	DB2EvmonUOW.xsl

You can create your own XSLT style sheet to transform XML documents. You can pass these style sheets into the Java-based tool using the `-ss stylesheet_name` option.

## Examples

### Example 1

To obtain a formatted text output for all events that have occurred in the last 32 hours from the package cache unformatted event table `PKG` in database `SAMPLE`, issue the following command:

```
java db2evmonfmt -d sample -ue pkg -ftext -hours 32
```

### Example 2

To obtain a formatted text output for all events of type `LOCKTIMEOUT` that have occurred in the last 24 hours from unformatted event table `LOCK` in database `SAMPLE`, issue the following command:

```
java db2evmonfmt -d sample -ue LOCK -ftext -hours 24 -type locktimeout
```

### Example 3

To obtain a formatted text output from the XML source file `LOCK.XML`, extracting all events that match the event type `LOCKWAIT` in the last 5 hours, issue the following command:

```
java db2evmonfmt -f lock.xml -ftext -type lockwait -hours 5
```

### Example 4

To obtain a formatted text output using the created XSLT style sheet `SUMMARY.XSL` for all events in the unformatted event table `UOW` in database `SAMPLE`, issue the following command:

```
java db2evmonfmt -d sample -ue uow -ftext -ss summary.xml
```

## Sample formatted flat-text output

The following sample of formatted flat-text output was generated from the locking event monitor XSLT style sheet:

```
-----  
Event Entry      : 0  
Event ID        : 1  
Event Type      : Locktimeout  
Event Timestamp : 2008-05-23-12.00.14.132329000  
-----
```

### Lock Details

```
-----  
Lock Name       : 020004010000000000000000054  
Lock Type      : Table  
Lock Attributes : 00000000  
Lock Count     : 1  
Lock Hold Count : 0  
Lock rrIID     : 0  
Lock Status    : Waiting  
Cursor Bitmap  : 00000000  
Tablespace Name : USERSPACE1  
Table Name     : NEWTON .SARAH
```

Attributes	Requestor	Holder
-----	-----	-----
Application Handle	[0-35]	[0-16]
Application ID	*LOCAL.horton.080523160016	*LOCAL.horton.080523155938
Application Name	xaplus0001	db2bp
Authentication ID	NEWTON	HORTON
Requesting Agent	65	21
Coordinating Agent	65	21
Application Status	SQLM_CONNECTPEND	SQLM_CONNECTPEND
Lock Timeout	5000	0
Workload Name	XAPLUS0010_WL02	SYSDEFAULTUSERWORKLOAD
Service Subclass	XAPLUS0010_SC02	SYSDEFAULTSUBCLASS
Current Request	Execute	Execute Immediate
Lock Mode	Intent Exclusive	Exclusive
tpmon Userid		
tpmon Wkstn		
tpmon App		
tpmon Accstring		

### Lock Requestor Current Activities

```
-----  
Activity ID     : 2  
Uow ID         : 1  
Package ID     : 65426E4D4B584659  
Package SectNo : 3  
Package Name   : NEWTON  
Package Schema : AKINTERF  
Package Version :  
Reopt         : always  
Eff Isolation  : Cursor Stability  
Eff Locktimeout : 5  
Eff Degree     : 0  
Nesting Level  : 0  
Stmt Unicode   : No  
Stmt Flag     : Dynamic  
Stmt Type     : DML, Insert/Update/Delete  
Stmt Text     : INSERT INTO SARAH VALUES(:H00008, :H00013, :H00014)
```

### Lock Requestor Past Activities

```

-----
Activity ID      : 1
Uow ID          : 1
Package ID      : 65426E4D4B584659
Package SectNo  : 2
Package Name    : NEWTON
Package Schema  : AKINTERF
Package Version :
Reopt          : always
Eff Isolation   : Cursor Stability
Eff Locktimeout : 5
Eff Degree      : 0
Nesting Level   : 0
Stmt Unicode    : No
Stmt Flag       : Dynamic
Stmt Type       : DML, Insert/Update/Delete
Stmt Text       : INSERT INTO NADIA VALUES(:H00007)

```

Lock Holder Current Activities

Lock Holder Past Activities

```

-----
Activity ID      : 1
Uow ID          : 2
Package ID      : 41414141414E4758
Package SectNo  : 201
Package Name    : NULLID
Package Schema  : SQLC2G13
Package Version :
Reopt          : none
Eff Isolation   : Cursor Stability
Eff Locktimeout : 5
Eff Degree      : 0
Nesting Level   : 0
Stmt Unicode    : No
Stmt Flag       : Dynamic
Stmt Type       : DML, Select (blockable)
Stmt Text       : select * from newton.sarah

```

```

Activity ID      : 2
Uow ID          : 2
Package ID      : 41414141414E4758
Package SectNo  : 203
Package Name    : NULLID
Package Schema  : SQLC2G13
Package Version :
Reopt          : none
Eff Isolation   : Cursor Stability
Eff Locktimeout : 5
Eff Degree      : 0
Nesting Level   : 0
Stmt Unicode    : No
Stmt Flag       : Dynamic
Stmt Type       : DML, Lock Table
Stmt Text       : lock table newton.sarah in exclusive mode

```

```

-----
Event Entry      : 1
Event ID         : 2
Event Type       : Locktimeout
Event Timestamp  : 2008-05-23-12.04.42.144896000
-----

```

...  
...  
...

## Usage notes

The db2evmonfmt utility is a Java-based tool which must be preceded by the java keyword in order to run successfully. The Java version required is that which is installed with the DB2 product from the sqllib/java/jdk64 directory.

**Note:** You can also use the EVMON\_FORMAT\_UE\_TO\_XML table function to format the binary events, contained in the unformatted event table BLOB column, into an XML document.

## Monitoring database locking

Diagnosing and correcting lock contention situations in large DB2 environments can be complex and time consuming. The lock event monitor and other facilities are designed to simplify this task by collecting locking data.

### Introduction

The lock event monitor is used to capture descriptive information about lock events at the time that they occur. The information captured identifies the key applications involved in the lock contention that resulted in the lock event. Information is captured for both the lock requestor (the application that received the deadlock or lock timeout error, or waited for a lock for more than the specified amount of time) and the current lock owner.

The information collected by the lock event monitor is written in binary format to an unformatted event table in the database. The captured data is processed in a post-capture step improving the efficiency of the capture process.

You can also directly access DB2 relational monitoring interfaces (table functions) to collect lock event information by using either dynamic or static SQL.

Determining if a deadlock or lock timeout has occurred is also simplified. Messages are written to the administration notification log when either of these events occurs; this supplements the SQL0911N (sqlcode -911) error returned to the application. In addition, a notification of lock escalations is also written to the administration notification log; this information can be useful in adjusting the size of the lock table and the amount of the table an application can use. There are also counters for lock timeouts (**lock\_timeouts**), lock waits (**lock\_waits**), and deadlocks (**deadlocks**) that can be checked.

The types of activities for which locking data can be captured include the following:

- SQL statements, such as:
  - DML
  - DDL
  - CALL
- LOAD command
- REORG command
- BACKUP DATABASE command

- Utility requests

The lock event monitor replaces the deprecated deadlock event monitors (CREATE EVENT MONITOR FOR DEADLOCKS statement and DB2DETAILDEADLOCK) and the deprecated lock timeout reporting feature (DB2\_CAPTURE\_LOCKTIMEOUT registry variable) with a simplified and consistent interface for gathering locking event data, and adds the ability to capture data on lock waits.

## Functional overview

Two steps are required to enable the capturing of lock event data using the locking event monitor:

1. You must create a LOCK EVENT monitor using the CREATE EVENT MONITOR FOR LOCKING statement. You provide a name for the monitor and the name of an unformatted event table into which the lock event data will be written.
2. You must specify the level for which you want lock event data captured by using one of the following methods:
  - You can specify particular workloads by either altering an existing workload, or by creating a new workload using the CREATE or ALTER WORKLOAD statements. At the workload level you must specify the type of lock event data you want captured (deadlock, lock timeout or lock wait), and whether you want the SQL statement history and input values for the applications involved in the locking. For lock waits you must also specify the amount of time that an application will wait for a lock, after which data is captured for the lock wait.
  - You can collect data at the database level and affect all DB2 workloads by setting the appropriate database configuration parameter:

### **mon\_lockwait**

This parameter controls the generation of lock wait events

Best practice is to enable lock wait data collection at the workload level.

### **mon\_timeout**

This parameter controls the generation of lock timeout events

Best practice is to enable lock timeout data collection at the database level if they are unexpected by the application. Otherwise enable at workload level.

### **mon\_deadlock**

This parameter controls the generation of deadlock events

Best practice is to enable deadlock data collection at the database level.

### **mon\_lw\_thresh**

This parameter controls the amount of time spent in lock wait before an event for **mon\_lockwait** is generated

The capturing of SQL statement history and input values incurs additional overhead, but this level of detail is often needed to successfully debug a locking problem.

After a locking event has occurred, the binary data in the unformatted event table can be transformed into an XML or a text document using a supplied Java-based application called `db2evmonfmt`. In addition, you can format the binary event data in the unformatted event table BLOB column into either an XML report document, using the `EVMON_FORMAT_UE_TO_XML` table function, or into a relational table, using the `EVMON_FORMAT_UE_TO_TABLES` procedure.

To aid in the determination of what workloads should be monitored for locking events, the administration notification log can be reviewed. Each time a deadlock or lock timeout is encountered, a message is written to the log. These messages identify the workload in which the lock requestor and lock owner or owners are running, and the type of locking event. There are also counters at the workload level for lock timeouts (**lock\_timeouts**), lock waits (**lock\_waits**), and deadlocks (**deadlocks**) that can be checked.

### Information collected for a locking event

Some of the information for lock events collected by the lock event monitor include the following:

- The lock that resulted in an event
- The application holding the lock that resulted in the lock event
- The applications that were waiting for or requesting the lock that result in the lock event
- What the applications were doing during the lock event

### Limitations

- There is no automatic purging of the lock event data written to the unformatted event table. You must periodically purge data from the table.
- You can output the collected event monitor data to only the unformatted event table. Outputs to file, pipe, and table are not supported.
- It is suggested that you create only one locking event monitor per database. Each additional event monitor only creates a copy of the same data.

### Deprecated lock monitoring functionality

The deprecated detailed deadlock event monitor, `DB2DETAILDEADLOCK`, is created by default for each database and starts when the database is activated. The `DB2DETAILDEADLOCK` event monitor must be disabled and removed, otherwise both the deprecated and new event monitors will be collecting data and will significantly affect performance.

To remove the `DB2DETAILDEADLOCK` event monitor, issue the following SQL statements:

```
SET EVENT MONITOR DB2DETAILDEADLOCK state 0
DROP EVENT MONITOR DB2DETAILDEADLOCK
```

### Collecting lock event data and generating reports

You can use the lock event monitor to collect lock timeout, lock wait, and deadlock information to help identify and resolve locking problems. After the lock event data has been collected in an unreadable form in an unformatted event table, this task describes how to obtain a readable text report.

### Before you begin

To create the locking event monitor and collect lock event monitor data, you must have DBADM, or SQLADM authority.

### About this task

The lock event monitor collects relevant information that helps with the identification and resolution of locking problems. For example, some of the information the lock event monitor collects for a lock event is as follows:

- The lock that resulted in a lock event
- The applications requesting or holding the lock that resulted in a lock event
- What the applications were doing during the lock event

This task provides instructions for collecting lock event data for a given workload. You might want to collect lock event data under the following conditions:

- You notice that lock wait values are longer than usual when using the `MON_GET_WORKLOAD` table function.
- An application returns a -911 SQL return code with reason code 68, stating that "The transaction was rolled back due to a lock timeout."
- You notice a deadlock event message in the administration notification log. The log message indicates that the lock event occurred between two applications, for example, Application A and B, where A is part of workload FINANCE and B is part of workload PAYROLL.

### Restrictions

To view data values, you need the EXECUTE privilege on the `EVMON_FORMAT_UE_*` routines, which the SQLADM and DBADM authorities hold implicitly. You also need SELECT privilege on the unformatted event table, which by default is held by users with the DATAACCESS authority and by the creator of the event monitor and the associated unformatted event table.

### Procedure

To collect detailed information regarding potential future lock events, perform the following steps:

1. Create a lock event monitor called `lockevmon` by using the CREATE EVENT MONITOR FOR LOCKING statement, as shown in the following example:

```
CREATE EVENT MONITOR lockevmon FOR LOCKING
WRITE TO UNFORMATTED EVENT TABLE
```

**Note:** The following lists important points to remember when creating an event monitor:

- You can create event monitors ahead of time and not worry about using up disk space since nothing is written until you activate the data collection at the database or workload level
- In a partitioned database environment, ensure that the event monitors are placed in a partitioned table space across all nodes. Otherwise, lock events will be missed at partitions where the partitioned table space is not present.
- Ensure that you set up a table space and bufferpool to minimize the interference on high performance work caused by ongoing work during accesses to the tables to obtain data.

2. Enable the lock event data collection at the workload level using the ALTER WORKLOAD statement with statement history. Setting the database configuration parameter affects the lock event data collection at the database level and all workloads are affected.

#### For lock wait events

To collect lock wait data for any lock acquired after 5 seconds for the FINANCE application and to collect lock wait data for any lock acquired after 10 seconds for the PAYROLL application, issue the following statements:

```
ALTER WORKLOAD finance COLLECT LOCK WAIT DATA WITH HISTORY AND VALUES
    FOR LOCKS WAITING MORE THAN 5 SECONDS
ALTER WORKLOAD payroll COLLECT LOCK WAIT DATA
    FOR LOCKS WAITING MORE THAN 10 SECONDS WITH HISTORY
```

To set the **mon\_lockwait** database configuration parameter with HIST\_AND\_VALUES input data value for the SAMPLE database, and to set the **mon\_lw\_thresh** database configuration parameter for 10 seconds, issue the following commands:

```
db2 update db cfg for sample using mon_lockwait hist_and_values
db2 update db cfg for sample using mon_lw_thresh 10000000
```

#### For lock timeout events

To collect lock timeout data for the FINANCE and PAYROLL applications, issue the following statements:

```
ALTER WORKLOAD finance COLLECT LOCK TIMEOUT DATA WITH HISTORY
ALTER WORKLOAD payroll COLLECT LOCK TIMEOUT DATA WITH HISTORY
```

To set the **mon\_locktimeout** database configuration parameter with HIST\_AND\_VALUES input data value for the SAMPLE database, issue the following command:

```
db2 update db cfg for sample using mon_locktimeout hist_and_values
```

#### For deadlock events

To collect data for the FINANCE and PAYROLL applications, issue the following statements:

```
ALTER WORKLOAD finance COLLECT DEADLOCK DATA WITH HISTORY
ALTER WORKLOAD payroll COLLECT DEADLOCK DATA WITH HISTORY
```

To set the **mon\_deadlock** database configuration parameter with HIST\_AND\_VALUES input data value for the SAMPLE database, issue the following command:

```
db2 update db cfg for sample using mon_deadlock hist_and_values
```

3. Rerun the workload in order to receive another lock event notification.
4. Connect to the database.
5. Obtain the locking event report using one of the following approaches:
  - a. Use the XML parser tool, db2evmonfmt, to produce a flat-text report based on the event data collected in the unformatted event table and using the default stylesheet, for example:

```
java db2evmonfmt -d db_name -ue table_name -ftext -u user_id -p password
```
  - b. Use the EVMON\_FORMAT\_UE\_TO\_XML table function to obtain an XML document.
  - c. Use the EVMON\_FORMAT\_UE\_TO\_TABLES procedure to output the data into a relational table.
6. Analyze the report to determine the reason for the lock event problem and resolve it.



- Turn OFF lock data collection for both FINANCE and PAYROLL applications by running the following statements or resetting the database configuration parameters:

**For lock wait events**

```
ALTER WORKLOAD finance COLLECT LOCK WAIT DATA NONE
ALTER WORKLOAD payroll COLLECT LOCK WAIT DATA NONE
```

To reset the **mon\_lockwait** database configuration parameter with the default NONE input data value for the SAMPLE database, and to reset the **mon\_lw\_thresh** database configuration parameter back to its default value of 5 seconds, issue the following command:

```
db2 update db cfg for sample using mon_lockwait none
db2 update db cfg for sample using mon_lw_thresh 5000000
```

**For lock timeout events**

```
ALTER WORKLOAD finance COLLECT LOCK TIMEOUT DATA NONE
ALTER WORKLOAD payroll COLLECT LOCK TIMEOUT DATA NONE
```

To reset the **mon\_locktimeout** database configuration parameter with the default NONE input data value for the SAMPLE database, issue the following command:

```
db2 update db cfg for sample using mon_locktimeout none
```

**For deadlock events**

```
ALTER WORKLOAD finance COLLECT DEADLOCK DATA NONE
ALTER WORKLOAD payroll COLLECT DEADLOCK DATA NONE
```

To reset the **mon\_deadlock** database configuration parameter with the default WITHOUT\_HIST input data value for the SAMPLE database, issue the following command:

```
db2 update db cfg for sample using mon_deadlock without_hist
```

**What to do next**

Rerun the application or applications to ensure that the locking problem has been eliminated.

**Information written to XML for locking event monitor**

Information written for a locking event monitor from the EVMON\_FORMAT\_UE\_TO\_XML table function. This is also documented in the sql1lib/misc/DB2EvmonLocking.xsd file.

**db2\_lock\_event**

The main schema that describes a lock timeout, lock wait or deadlock event in detail.

Table 4. db2\_lock\_event

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
db2_lock_event	db2lockevent	1	1	The main schema that describes a lock timeout, lock wait or deadlock event in details.

## db2lockevent

This schema describes the structure of each locking event captured by the event monitor.

Table 5. db2lockevent attributes

Name	Data Type	Use	Description
id	xs:positiveInteger	required	Integer representing the Event ID.
type	xs:string	required	Type of locking event that has occurred. Lock events can be of the following types: Locktimeout, Deadlock, or Lockwait.
timestamp	db2_timestamp_type	required	Timestamp representing when the lock event occurred.
member	member_type	required	Member where the lock event occurred.
release	xs:nonNegativeInteger	required	Represents the DB2 product level this event was captured on.

The following elements will be returned for a normal event:

Table 6. db2lockevent elements returned for a normal event

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
db2_deadlock_graph	db2dgraph	0	1	Schema element represents the DB2 Deadlock Graph. The graph outlines all the participants involved in the deadlock.
db2_participant	db2appinfo	1	unbounded	Schema element represents the application information of the all the participants involved in a lock event.

If an error condition occurs, the following elements are returned giving message details:

Table 7. db2lockevent elements returned for an error condition

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
db2_message	xs:string	1	1	Error message
db2_event_file	xs:string	1	1	Fully qualified path to file where event has been written.

## db2appdetails

This schema represents the details regarding the participant.

Table 8. db2appdetails elements

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
application_handle	application_handle_type	1	1	A system-wide unique ID for the application. See monitor element agent_id for more details.
appl_id	appl_id_type	1	1	This identifier is generated when the application connects to the database at the database manager. See monitor element appl_id for more details.
appl_name	xs:string	1	1	The name of the application running at the client, as known to the database. See monitor element appl_name for more details.
auth_id	authid_type	1	1	The authorization ID of the user who invoked the application that is being monitored. See monitor element auth_id for more details.
agent_tid	xs:nonNegativeInteger	1	1	The unique identifier for the engine dispatchable unit (EDU) for the agent. See monitor element agent_pid for more details.
coord_agent_tid	xs:nonNegativeInteger	1	1	The engine dispatchable unit (EDU) identifier of the coordinator agent for the application. See monitor element coord_agent_pid for more details.
appl_status	.	1	1	The current status of the application. See monitor element appl_status for more details.
appl_action	.	1	1	The action/request that the client application is performing
lock_timeout_val	xs:integer	1	1	The database configuration parameter lock timeout. Value in seconds. See monitor element lock_timeout_val for more details.

Table 8. db2appdetails elements (continued)

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
lock_wait_val	xs:integer	1	1	The lock wait parameter in effect during the lock event. This is either the database configuration parameter MON_LW_THRESH or the COLLECT LOCK WAIT DATA setting specified at the workload level. Value in milliseconds.
tentry_state	.	1	1	TEntry state. Internal use only.
tentry_flag1	xs:hexBinary	1	1	TEntry flags1. Internal use only.
tentry_flag2	xs:hexBinary	1	1	TEntry flags2. Internal use only.
xid	xs:hexBinary	1	1	XID - Global transaction identifier
workload_id	db_object_id_type	1	1	ID of the workload to which this application belongs. See monitor element workload_id for more details.
workload_name	db_object_name_type	1	1	Name of the workload to which this application belongs. See monitor element workload_name for more details.
service_class_id	db_object_id_type	1	1	ID of the service subclass to which this application belongs. See monitor element service_class_id for more details.
service_subclass_name	db_object_name_type	1	1	Name of the service subclass to which this application belongs. See monitor element service_subclass_name for more details.
current_request	xs:string	1	1	The operation currently being processed or most recently processed.
lock_escalation	xs:string	1	1	Indicates whether a lock request was made as part of a lock escalation. See monitor element lock_escalation for more details. Possible values: Yes or No.

Table 8. db2appdetails elements (continued)

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
past_activities_wrapped	xs:string	1	1	Indicates whether the activities list has wrapped. The default limit on the number of past activities to be kept by any one application is 250. This default can be overridden using the registry variable DB2_MAX_INACT_STMTS. Users may want to choose a different value for the limit to increase or reduce the amount of system monitor heap used for inactive statement information.
client_userid	tpmon_type	1	1	The client user ID generated by a transaction manager and provided to the server. See monitor element client_userid for more details.
client_wrkstnname	tpmon_type	1	1	Identifies the client system or workstation, if the sqlesei API was issued in this connection. See monitor element client_wrkstnname for more details.
client_applname	tpmon_type	1	1	Identifies the server transaction program performing the transaction, if the sqlesei API was issued in this connection. See monitor element client_applname for more details.
client_acctng	tpmon_type	1	1	The data passed to the target database for logging and diagnostic purposes, if the sqlesei API was issued in this connection. See monitor element client_acctng for more details.

### db2appinfo

This schema describes the structure of the application that is either requesting the lock or holding the lock.

Table 9. db2appinfo attributes

Name	Data Type	Use	Description
no	xs:positiveInteger	required	A sequence number uniquely identifying this participant in a lock event.
type	xs:string	required	The participant type. The participant can either be the Requestor or Owner.
participant_no_holding_lk	xs:positiveInteger	optional	The participant number identifying the application holding the lock.

Table 10. db2appinfo elements

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
db2_object_requested	db2objectrequested	0	1	Schema element represents the DB2 lock that the Requestor is attempting to acquire, which is being held by the Owner.
db2_app_details	db2appdetails	1	1	Schema element represents the details regarding this participant.
db2_activity	db2activity	0	unbounded	List of all DB2 activities the application is currently executing or has executed.

### db2objectrequested

Table 11. db2objectrequested attributes

Name	Data Type	Use	Description
type	xs:string	required	The type of object being requested. Possible values: lock or ticket.

If the attribute db2objectrequested/type is the value of lock, then the following elements will be returned:

Table 12. db2objectrequested elements returned for locks

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
lock_name	xs:hexBinary	1	1	Internal binary lock name. This element serves as a unique identifier for locks. See monitor element lock_name for more details.
lock_object_type	.	1	1	The type of object the application is waiting to obtain a lock. See monitor element lock_object_type for more details.

Table 12. db2objectrequested elements returned for locks (continued)

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
lock_specifics	xs:string	1	1	Internal specifics about the lock. For information use only.
lock_attributes	xs:hexBinary	1	1	Lock attributes. See monitor element lock_attributes for more details.
lock_current_mode	.	1	1	Original lock before conversion. See monitor element lock_current_mode for more details.
lock_mode_requested	.	1	1	The lock mode being requested by this participant. See monitor element lock_mode_requested for more details.
lock_mode	.	1	1	The type of lock being held. See monitor element lock_mode for more details.
lock_count	xs:integer	1	1	The number of locks on the lock being held. See monitor element lock_count for more details.
lock_hold_count	xs:integer	1	1	The number of holds placed on the lock. See monitor element lock_hold_count for more details.
lock_rriid	xs:integer	1	1	IID for Row locking. Internal use only.
lock_status	.	1	1	Indicates the internal status of the lock. See monitor element lock_status for more details.
lock_release_flags	xs:hexBinary	1	1	Lock release flags. See monitor element lock_release_flags for more details.
tablespace_name	.	1	1	The name of the table space where the lock is held. See monitor element tablespace_name for more details.
table_name	.	1	1	The name of the table where the lock is held. See monitor element table_name for more details.

Table 12. db2objectrequested elements returned for locks (continued)

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
table_schema	db_object_name_type	1	1	The schema of the table. See monitor element table_schema for more details.

If the attribute db2objectrequested/type is the value of ticket, then the following elements will be returned:

Table 13. db2objectrequested elements returned for tickets

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
threshold_name	db_object_name_type	1	1	The name of the threshold queue.
threshold_id	db_object_id_type	1	1	The ID of the threshold queue.
queued_agents	xs:nonNegativeInteger	1	1	The total number of agents currently queued in the threshold.

## db2dgraph

This schema describes the structure of the deadlock graph and the participants involved in the deadlock cycle.

Table 14. db2dgraph attributes

Name	Data Type	Use	Description
dl_conns	xs:nonNegativeInteger	required	The total number of participants involved in a deadlock. See monitor element dl_conns for more details.
rolled_back_participant_no	xs:nonNegativeInteger	required	The participant number identifying the rolled back application. See monitor element rolled_back_participant_no for more details.
type	xs:string	required	Type of deadlock that has occurred: local or global. In a global deadlock at least one or more participants reside on a different members. In a local deadlock all participants reside on the same member.



Table 15. db2dgraph elements

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
db2_participant	db2dstack	1	unbounded	Schema element represents a single stack entry in a deadlock graph.

### db2dstack

This schema describes the structure of an entry in a deadlock graph. The entry contains information regarding the participant requesting and holding the lock in a deadlock cycle.

Table 16. db2dstack attributes

Name	Data Type	Use	Description
no	xs:positiveInteger	required	The participant number identifying the application requesting the lock.
deadlock_member	member_type	required	Member where the participant is requesting the lock.
participant_no_holding_lk	xs:positiveInteger	required	The participant number identifying the application holding the lock.
application_handle	application_handle_type	required	A system-wide unique ID for the application. See monitor element agent_id for more details.

### db2actdetails

This schema describes the details regarding an activity.

Table 17. db2actdetails elements

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
activity_id	xs:positiveInteger	1	1	Counter which uniquely identifies an activity for an application within a given unit of work. See monitor element activity_id for more details.
uow_id	xs:positiveInteger	1	1	The unit of work ID to which this activity record applies. See monitor element uow_id for more details.
package_name	xs:string	1	1	The name of the package that contains the SQL statement currently executing. See monitor element package_name for more details.

Table 17. db2actdetails elements (continued)

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
package_schema	xs:string	1	1	The authorization ID of the user that precompiled the application. See monitor element package_schema for more details.
package_version_id	xs:string	1	1	The package version identifies the version identifier of the package that contains the SQL statement currently executing. See monitor element package_version_id for more details.
consistency_token	xs:string	1	1	The package consistency token helps to identify the version of the package that contains the SQL statement currently executing. See monitor element consistency_token for more details.
section_number	xs:nonNegativeInteger	1	1	The internal section number in the package for the SQL statement currently processing or most recently processed. See monitor element section_number for more details.
reopt	xs:string	1	1	The REOPT bind option used to precompile this package. Possible values are: NONE, ONCE, and ALWAYS. See the REOPT bind options for more details.
incremental_bind	xs:string	1	1	The package was incrementally bound at execution time. Possible values: Yes or No.
effective_isolation	.	1	1	The isolation value in effect for the SQL statement while it was being run. See monitor element effective_isolation for more details.
effective_query_degree	xs:nonNegativeInteger	1	1	The degree value in effect for the SQL statement while it was being run. See monitor element effective_query_degree for more details.

Table 17. db2actdetails elements (continued)

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
stmt_unicode	xs:string	1	1	The SQL statement unicode flag. Possible values: Yes or No.
stmt_lock_timeout	xs:integer	1	1	The locktimeout value in effect for the SQL statement while it was being run. See monitor element stmt_lock_timeout for more details
stmt_type	xs:string	1	1	The type of SQL statement processed. Possible values: Dynamic or Static. See monitor element stmt_type for more details.
stmt_operation	xs:string	1	1	The SQL statement operation type. See monitor element stmt_operation for more details.
stmt_query_id	xs:nonNegativeInteger	1	1	Internal query identifier given to any SQL statement. See monitor element stmt_query_id for more details.
stmt_nest_level	xs:nonNegativeInteger	1	1	This element shows the level of nesting or recursion in effect when the statement was run. See monitor element stmt_nest_level for more details.
stmt_invocation_id	xs:nonNegativeInteger	1	1	This element shows the identifier of the routine invocation in which the SQL statement was run. See monitor element stmt_invocation_id for more details.
stmt_source_id	xs:nonNegativeInteger	1	1	This element shows the internal identifier given to the source of the SQL statement that was run. See monitor element stmt_source_id for more details.
stmt_pkgcache_id	xs:nonNegativeInteger	1	1	This element shows the internal package cache identifier of a dynamic SQL statement. See monitor element stmt_pkgcache_id for more details.

Table 17. db2actdetails elements (continued)

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
stmt_text	xs:string	1	1	The text of the SQL statement. See monitor element stmt_text for more details.

### db2activity

This schema describes the structure of a single DB2 activity for an application in a given unit of work.

Table 18. db2activity attributes

Name	Data Type	Use	Description
type	xs:string	required	The attribute represents the type or activity. Possible values are current or past.

Table 19. db2activity elements

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
db2_activity_details	db2actdetails	1	1	Schema represents the details regarding this activity
db2_input_variable	db2inputvar	0	unbounded	Schema element represents the list of input variables associated with the SQL statement.

### db2inputvar

This schema describes the structure of a single input variable.

Table 20. db2inputvar elements

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
stmt_value_index	xs:positiveInteger	1	1	The element represents the position of the input parameter marker or host variable used in the SQL statement. See monitor element stmt_value_index for more details.
stmt_value_isreopt	xs:string	1	1	The element shows whether the variable was used during statement reoptimization. See monitor element stmt_value_isreopt for more details.

Table 20. db2inputvar elements (continued)

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
stmt_value_isnull	xs:string	1	1	The element shows whether a data value associated with the SQL statement is the NULL value. See monitor element stmt_value_isnull for more details.
stmt_value_type	xs:string	1	1	The element contains a string representation of the type of data value associated with an SQL statement. See monitor element stmt_value_type for more details.
stmt_value_data	xs:string	1	1	The element contains a string representation of a data value associated with an SQL statement. See monitor element stmt_value_data for more details.

### Information written to relational table for locking event monitor

Information written for a locking event monitor from an XML document to a relational table using the EVMON\_FORMAT\_UE\_TO\_TABLES procedure. This is also documented in the sqllib/misc/DB2EvmonLocking.xsd file.

### Information written for a locking event monitor

Table 21. Information returned for a locking event monitor: Table name: LOCK\_EVENT

Column Name	Data Type	Description
XMLID	VARCHAR(1024) NOT NULL	
EVENT_ID	BIGINT NOT NULL	
EVENT_TYPE	VARCHAR(128) NOT NULL	
EVENT_TIMESTAMP	TIMESTAMP NOT NULL	
MEMBER	SMALLINT NOT NULL	member - Database member
DL_CONNS	BIGINT	dl_conns - Connections involved in deadlock
ROLLED_BACK_PARTICIPANT_NO	BIGINT	rolled_back_participant_no - Rolled back application participant

Table 22. Information returned for a locking event monitor: Table name: LOCK\_PARTICIPANTS

Column Name	Data Type	Description
XMLID	VARCHAR(1024) NOT NULL	
PARTICIPANT_NO	BIGINT	participant_no - Participant within deadlock
PARTICIPANT_TYPE	VARCHAR(10)	

Table 22. Information returned for a locking event monitor: Table name: LOCK\_PARTICIPANTS (continued)

Column Name	Data Type	Description
PARTICIPANT_NO_HOLDING_LK	BIGINT	participant_no_holding_lk - Participant holding a lock on the object required by application
APPLICATION_HANDLE	BIGINT	application_handle - Application handle
APPL_ID	VARCHAR(128)	appl_id - Application ID
APPL_NAME	VARCHAR(128)	appl_name - Application name
AUTH_ID	VARCHAR(128)	auth_id - Authorization ID
AGENT_TID	BIGINT	agent_pid - Engine dispatchable unit (EDU) identifier
COORD_AGENT_TID	BIGINT	coord_agent_pid - Coordinator agent identifier
APPL_STATUS	BIGINT	appl_status - Application status
LOCK_TIMEOUT_VAL	BIGINT	lock_timeout_val - Lock timeout value
LOCK_WAIT_VAL	BIGINT	
WORKLOAD_ID	BIGINT	workload_id - Workload ID
WORKLOAD_NAME	VARCHAR(128)	workload_name - Workload name
SERVICE_CLASS_ID	BIGINT	service_class_id - Service class ID
SERVICE_SUBCLASS_NAME	VARCHAR(128)	service_subclass_name - Service subclass name
CURRENT_REQUEST	VARCHAR(32)	
LOCK_ESCALATION	CHAR(3)	lock_escalation - Lock escalation
PAST_ACTIVITIES_WRAPPED	CHAR(3)	
CLIENT_USERID	VARCHAR(64)	
CLIENT_WRKSTNNAME	VARCHAR(128)	
CLIENT_APPLNAME	VARCHAR(128)	
CLIENT_ACCTNG	VARCHAR(200)	
OBJECT_REQUESTED	VARCHAR(10)	
LOCK_NAME	CHAR(26)	lock_name - Lock name
LOCK_OBJECT_TYPE	BIGINT	lock_object_type - Lock object type waited on
LOCK_ATTRIBUTES	CHAR(8)	lock_attributes - Lock attributes
LOCK_CURRENT_MODE	BIGINT	lock_current_mode - Original lock mode before conversion
LOCK_MODE_REQUESTED	BIGINT	lock_mode_requested - Lock mode requested
LOCK_MODE	BIGINT	lock_mode - Lock mode
LOCK_COUNT	BIGINT	lock_count - Lock count
LOCK_HOLD_COUNT	BIGINT	lock_hold_count - Lock hold count
LOCK_RRIID	BIGINT	
LOCK_STATUS	VARCHAR(10)	lock_status - Lock status
LOCK_RELEASE_FLAGS	CHAR(8)	lock_release_flags - Lock release flags

Table 22. Information returned for a locking event monitor: Table name: LOCK\_PARTICIPANTS (continued)

Column Name	Data Type	Description
TABLE_FILE_ID	BIGINT	table_file_id - Table file identification
TABLE_NAME	VARCHAR(128)	table_name - Table name
TABLE_SCHEMA	VARCHAR(128)	table_schema - Table schema name
TABLESPACE_NAME	VARCHAR(128)	tablespace_name - Table space name
THRESHOLD_ID	BIGINT	
THRESHOLD_NAME	VARCHAR(128)	threshold_name - Threshold name

Table 23. Information returned for a locking event monitor: Table name: LOCK\_PARTICIPANT\_ACTIVITIES

Column Name	Data Type	Description
XMLID	VARCHAR(1024) NOT NULL	
PARTICIPANT_NO	BIGINT	participant_no - Participant within deadlock
ACTIVITY_ID	BIGINT	activity_id - Activity ID
ACTIVITY_TYPE	VARCHAR(10)	activity_type - Activity type
UOW_ID	BIGINT	uow_id - Unit of work ID
PACKAGE_NAME	VARCHAR(128)	package_name - Package name
PACKAGE_SCHEMA	VARCHAR(128)	package_schema - Package schema
PACKAGE_VERSION_ID	VARCHAR(64)	package_version_id - Package version
CONSISTENCY_TOKEN	VARCHAR(8)	consistency_token - Package consistency token
SECTION_NUMBER	BIGINT	section_number - Section number
REOPT	VARCHAR(10)	
INCREMENTAL_BIND	CHAR(3)	
EFFECTIVE_ISOLATION	BIGINT	effective_isolation - Effective isolation
EFFECTIVE_QUERY_DEGREE	BIGINT	effective_query_degree - Effective query degree
STMT_LOCK_TIMEOUT	BIGINT	stmt_lock_timeout - Statement lock timeout
STMT_TYPE	VARCHAR(10)	stmt_type - Statement type
STMT_QUERY_ID	BIGINT	stmt_query_id - Statement query identifier
STMT_NEST_LEVEL	SMALLINT	stmt_nest_level - Statement nesting level
STMT_INVOCATION_ID	BIGINT	stmt_invocation_id - Statement invocation identifier
STMT_SOURCE_ID	BIGINT	stmt_source_id - Statement source identifier
STMT_PKG_CACHE_ID	BIGINT	stmt_pkgcache_id - Statement package cache identifier
STMT_TEXT	CLOB(2097152)	stmt_text - SQL statement text

Table 24. Information returned for a locking event monitor: Table name: LOCK\_ACTIVITY\_VALUES

Column Name	Data Type	Description
XMLID	VARCHAR(1024) NOT NULL	
PARTICIPANT_NO	BIGINT	participant_no - Participant within deadlock
ACTIVITY_ID	BIGINT	activity_id - Activity ID
UOW_ID	BIGINT	uow_id - Unit of work ID
STMT_VALUE_INDEX	BIGINT	stmt_value_index - Value index
STMT_VALUE_ISREOPT	CHAR(3)	stmt_value_isreopt - Variable used for statement reoptimization
STMT_VALUE_ISNULL	CHAR(3)	stmt_value_isnull - Value has null value
STMT_VALUE_TYPE	CHAR(16)	stmt_value_type - Value type
STMT_VALUE_DATA	CLOB(32K)	stmt_value_data - Value data

## Monitoring unit of work events

The unit of work (UoW) event monitor records an event whenever a unit of work is completed, that is, whenever there is a commit or a rollback. This historical information about individual UoWs is useful for chargeback purposes (charging by CPU usage) and for monitoring compliance with response time service level objectives.

The UoW event monitor is one way to perform system perspective monitoring with request metrics. The most closely related alternatives or complements to the unit of work event monitor are either the statistics event monitor or the MON\_GET\_UNIT\_OF\_WORK and MON\_GET\_UNIT\_OF\_WORK\_DETAILS table functions.

To create the UoW event monitor and collect lock event monitor data, you must have DBADM or SQLADM authority.

### Creating a UoW event monitor and configuring data collection

Before you create a UoW event monitor, identify the table space where you plan to store the unformatted event table for your event monitor. The recommended practice is to have a table space dedicated and configured to store the unformatted event table associated with any event monitor. Create the unit of work event monitor in a tablespace with at least 8K pagesize to ensure that the event data is contained within the inlined BLOB column of the unformatted event table. If the BLOB column is not inlined, then the performance of writing and reading the events to the unformatted event table might not be efficient.

The database manager attempts to inline the event\_data BLOB column in the unformatted event table, but this is not always possible. To check that the rows in the unformatted event table have been inlined, use the ADMIN\_IS\_INLINED function. If the rows have not been inlined, use the ADMIN\_EST\_INLINE\_LENGTH functions to determine how much space the rows need.

Your other options when you create an event monitor are to specify any existing table space or to not specify any and have one chosen by default.



To setup a UoW event monitor using defaults and best practices, complete the following steps:

1. Create the event monitor by issuing the CREATE EVENT MONITOR statement. The following example uses defaults where possible and specifies to store the unformatted event table in an existing table space:

```
CREATE EVENT MONITOR MY_UOW_EVMON
  FOR UNIT OF WORK
  WRITE TO UNFORMATTED EVENT TABLE (IN MY_EVMON_TABLESPACE)
```

2. Configure what data to collect. The following statement illustrates a simple approach:

```
db2 update db cfg for dbname using mon_uow_data base
```

## Configuring data collection

To configure data collection, you must also specify the subset of the system workload for which to capture events and how much detail to collect for each event. By default UoW data is not collected. You can change the default settings by using one of the following settings:

- The **mon\_uow\_data** database configuration parameter
- The COLLECT UNIT OF WORK DATA clause of the CREATE/ALTER WORKLOAD statement

The following levels for data collection are available to you:

**None** No UoW data collected

**Base** UoW data collected

If either the **mon\_uow\_data** database configuration parameter or the COLLECT UNIT OF WORK DATA clause of the CREATE/ALTER WORKLOAD statement is set to BASE, then that is the effective setting for the workload.

If you want to enable data collection for only selected workloads, then set **mon\_uow\_data** database configuration parameter to NONE and set the level to BASE for the desired workloads.

Requests metrics is one of the types of information that you can collect with a UoW event monitor. The UoW event monitor is one of the interfaces affected by the setting for request metric collection. By default, request metrics are collected and reported in applicable table functions and event monitors, including the UoW event monitor. You can change the default setting by using one of the following settings:

- The **mon\_req\_metrics** database configuration parameter
- The COLLECT REQUEST METRICS clause of the CREATE/ALTER SERVICE CLASS statement for a service superclass.

Changing these settings affects any table function or event monitor that can report request metrics.

## Accessing event data captured by a UoW event monitor

This type of event monitor writes data in a binary format to an unformatted event table. You can access this data using the following table functions:

- **EVMON\_FORMAT\_UE\_TO\_XML** - extracts data from an unformatted event table into an XML document.

- `EVMON_FORMAT_UE_TO_TABLES` - extracts data from an unformatted event table into a set of relational tables.

Use these table functions to specify the data to extract using a `SELECT` statement. You have full control over selection, ordering, and other aspects provided by the `SELECT` statement.

You can also use the `db2evmonfmt` command to perform the following tasks:

- Select events of interest based on the following attributes: event ID, event type, time period, application, workload, or service class.
- Choose whether to receive the output in the form of a text report or a formatted XML document.
- Control the output format by creating your own XSLT style sheets instead of using the ones provided by the `db2evmonfmt` command.

For example, the following command provides a UoW report that:

1. Selects UoW events that have occurred in the past 24 hours in the database `SAMPLE`. These event records are obtained from the unformatted event table called `SAMPLE_UoW_events`.
2. Provides formatted text output using the `DB2EvmonUOW.xsl` style sheet.

```
java db2evmonfmt -d SAMPLE -ue SAMPLE_UOW_EVENTS -ftext -ss DB2EvmonUOW.xsl -hours 24
```

## Collecting unit of work event data and generating reports

You can use the unit of work event monitor to collect data about transactions that can be used for chargeback purposes. After the transaction event data has been collected in an unreadable form in an unformatted event table, this task describes how to obtain a readable text report.

### Before you begin

To collect unit of work event monitor data, you must have `SYSADM` or `SYSCTRL` authority.

### About this task

The unit of work event monitor collects relevant information that identifies application transactions and the corresponding CPU usage that can be used for chargeback purposes. For example, some of the information the unit of work event monitor collects for a transaction event is as follows:

- Total CPU usage time (`TOTAL_CPU_TIME`)
- Application handle (`APPLICATION_HANDLE`)

This task provides instructions for collecting unit of work event data for a given workload.

### Restrictions

Input data values are not viewable if you do not have `SYSADM` or `SYSCTRL` authority.

### Procedure

To collect detailed information regarding unit of work events, perform the following steps:

1. Create a unit of work event monitor called uowevmon by using the CREATE EVENT MONITOR FOR UNIT OF WORK statement, as shown in the following example:

```
CREATE EVENT MONITOR uowevmon FOR UNIT OF WORK
WRITE TO UNFORMATTED EVENT TABLE
```

2. Enable the unit of work event data collection at the workload level using the ALTER WORKLOAD statement with statement history. To collect unit of work data for the FINANCE and PAYROLL applications, issue the following statements:

```
ALTER WORKLOAD finance COLLECT UNIT OF WORK DATA WITH HISTORY
ALTER WORKLOAD payroll COLLECT UNIT OF WORK DATA WITH HISTORY
```

3. Rerun the workload in order to collect unit of work transaction events.

4. Connect to the database.

5. Obtain the unit of work event report using the following approach:

- a. Use the XML parser tool, db2evmonfmt, to produce a flat-text report based on the event data collected in the unformatted event table, for example:

```
java db2evmonfmt -d db_name -ue table_name -ftext -u user_id -p password
```

6. Analyze the report to determine how much CPU time applications are using so that appropriate charges can be billed.

7. If you want to turn OFF unit of work data collection for both FINANCE and PAYROLL applications, run the following statements:

```
ALTER WORKLOAD finance COLLECT UNIT OF WORK DATA NONE
ALTER WORKLOAD payroll COLLECT UNIT OF WORK DATA NONE
```

### Information written to XML for a unit of work event monitor

Information written for a unit of work event monitor from the EVMON\_FORMAT\_UE\_TO\_XML table function. This is also documented in the sql1lib/misc/DB2EvmonUOW.xsd file.

### Information written for a unit of work event monitor

**db2\_uow\_event:** The main schema that describes a unit of work event

Table 25. Elements

Name	Data type	Description
db2_uow_event	db2uowevent	The main schema that describes a unit of work event

### db2uowevent: the main schema that describes a unit of work event

Table 26. Attributes

Name	Data type	Description
id	xs:positiveInteger	Integer representing the Event ID
type	xs:string	Type of unit of work event that has occurred. Events can be of the following types: UOW
timestamp	db2_timestamp_type	Timestamp representing when the UOW event occurred
member	member_type	Member where the UOW event occurred
release	xs:nonNegativeInteger	Represents the DB2 product level this event was captured on

Table 27. Elements

Name	Data type	Minimum occurrence	Maximum occurrence	Description
completion_status	xs:string	1	1	The completion status of the unit of work. Possible values are: UNKNOWN, COMMIT, ROLLBACK, GLOBAL_COMMIT, GLOBAL_ROLLBACK, XA_END, XA_PREPARE
start_time	xs:dateTime	1	1	The start time of the unit of work. See monitor element uow_start_time for more details
stop_time	xs:dateTime	1	1	The stop time of the unit of work. See monitor element uow_stop_time for more details
connection_time	xs:dateTime	1	1	The time the application connected to the database member. See monitor element conn_time for more details
application_name	xs:string	1	1	The name of the application running at the client, as known to the database. See monitor element appl_name for more details
application_handle	application_handle_type	1	1	A system-wide unique ID for the application. See monitor element agent_id for more details
application_id	appl_id_type	1	1	This identifier is generated when the application connects to the database at the database manager. See monitor element appl_id for more details
uow_id	incrementing_id_type	1	1	The unit of work ID to which this activity record applies. See monitor element uow_id for more details
workload_occurrence_id	incrementing_id_type	1	1	The workload occurrence ID to which this activity record applies. See monitor element workload_occurrence_id for more details
coord_member	member_type	1	1	The coordinating member for this unit of work. See monitor element coord_partition_num for more details

Table 27. Elements (continued)

Name	Data type	Minimum occurrence	Maximum occurrence	Description
member_activation_time	xs:dateTime	1	1	The time this database member was activated. See monitor element db_conn_time for more details
workload_name	db_object_name_type	1	1	The name of the workload under which the unit of work completed. See monitor element workload_name for more details
workload_id	db_object_id_type	1	1	The workload ID of the workload under which the unit of work completed. See monitor element workload_id for more details
service_superclass_name	db_object_name_type	0	1	The name of the service super class under which the unit of work completed. See monitor element service_superclass_name for more details
service_subclass_name	db_object_name_type	0	1	The name of the service sub class under which the unit of work completed. See monitor element service_subclass_name for more details
service_class_id	db_object_id_type	0	1	The service class ID of the service class under which the unit of work completed. See monitor element service_class_id for more details
session_authid	authid_type	0	1	The session authorization ID of the user who invoked the application that is being monitored. See monitor element auth_id for more details
system_authid	authid_type	1	1	The system authorization ID of the user who invoked the application that is being monitored. See monitor element auth_id for more details
client_pid	xs:nonNegativeInteger	1	1	The process ID reported by the client. See monitor element client_pid for more details

Table 27. Elements (continued)

Name	Data type	Minimum occurrence	Maximum occurrence	Description
client_product_id	xs:string	1	1	The product ID of the client. See monitor element client_prdid for more details
client_platform	xs:nonNegativeInteger	1	1	The platform of the client. See monitor element client_platform for more details
client_protocol	xs:string	0	1	The product ID of the client. See monitor element client_protocol for more details
client_userid	tpmon_type	0	1	The client user ID generated by a transaction manager and provided to the server. See monitor element client_userid for more details
client_wrkstnname	tpmon_type	0	1	Identifies the client system or workstation, if the sqleseti API was issued in this connection. See monitor element client_wrkstnname for more details
client_applname	tpmon_type	0	1	Identifies the server transaction program performing the transaction, if the sqleseti API was issued in this connection. See monitor element client_applname for more details
client_acctng	tpmon_type	0	1	The data passed to the target database for logging and diagnostic purposes, if the sqleseti API was issued in this connection. See monitor element client_acctng for more details
local_transaction_id	xs:hexBinary	1	1	The local transaction ID for the unit of work
global_transaction_id	xs:hexBinary	1	1	The global transaction ID for the unit of work
system_metrics	system_level_metrics	1	1	The metrics for the unit of work. This is an XML element that contains all of the system level metrics.

## Information written to relational table for unit of work event monitor

Information written for a unit of work event monitor from an XML document to a relational table using the EVMON\_FORMAT\_UE\_TO\_TABLES procedure. This is also documented in the sql1lib/misc/DB2EvmonUOW.xsd file.

## Information written for an UOW event monitor

Table 28. Information returned for an UOW event monitor: Table name: UOW\_EVENT

Column Name	Data Type	Description
EVENT_ID	INTEGER NOT NULL	
TYPE	VARCHAR(128) NOT NULL	
EVENT_TIMESTAMP	TIMESTAMP NOT NULL	
MEMBER	SMALLINT	member - Database member
COORD_MEMBER	SMALLINT	coord_member - Coordinator member
COMPLETION_STATUS	VARCHAR(128)	
START_TIME	TIMESTAMP	start_time - Event start time
STOP_TIME	TIMESTAMP	stop_time - Event stop time
WORKLOAD_NAME	VARCHAR(128)	workload_name - Workload name
WORKLOAD_ID	INTEGER	workload_id - Workload ID
SERVICE_SUPERCLASS_NAME	VARCHAR(128)	service_superclass_name - Service superclass name
SERVICE_SUBCLASS_NAME	VARCHAR(128)	service_subclass_name - Service subclass name
SERVICE_CLASS_ID	INTEGER	service_class_id - Service class ID
UOW_ID	INTEGER	uow_id - Unit of work ID
WORKLOAD_OCCURRENCE_ID	INTEGER	workload_occurrence_id - Workload occurrence identifier
CONNECTION_TIME	TIMESTAMP	
MEMBER_ACTIVATION_TIME	TIMESTAMP	
APPLICATION_ID	VARCHAR(128)	
APPLICATION_HANDLE	BIGINT	application_handle - Application handle
APPLICATION_NAME	VARCHAR(128)	
SYSTEM_AUTHID	VARCHAR(128)	
SESSION_AUTHID	VARCHAR(128)	
CLIENT_PLATFORM	INTEGER	client_platform - Client operating platform
CLIENT_PID	INTEGER	client_pid - Client process ID
CLIENT_PRODUCT_ID	VARCHAR(128)	
CLIENT_PROTOCOL	VARCHAR(10)	client_protocol - Client communication protocol
CLIENT_WRKSTNNAME	VARCHAR(128)	
CLIENT_ACCTNG	VARCHAR(200)	
CLIENT_USERID	VARCHAR(64)	
CLIENT_APPLNAME	VARCHAR(128)	

Table 28. Information returned for an UOW event monitor: Table name: UOW\_EVENT (continued)

Column Name	Data Type	Description
LOCAL_TRANSACTION_ID	VARCHAR(16)	
GLOBAL_TRANSACTION_ID	VARCHAR(40)	
METRICS	BLOB(100K)	XML document that contains all of the system level metrics.

## Capturing system monitor elements using the statistics event monitor

The statistics event monitor contains the details\_xml monitor element in the event\_scstats and event\_wlstats logical data groups. Use this monitor element to capture information about the system.

The monitor element details\_xml is an XML document containing all the system monitor elements reported by the MON\_GET\_SERVICE\_SUBCLASS\_DETAILS and MON\_GET\_WORKLOAD\_DETAILS table functions. System monitor elements are a subset of the details document reported in the DETAILS column of the MON\_GET\_SERVICE\_SUBCLASS\_DETAILS and MON\_GET\_WORKLOAD\_DETAILS table functions.

Request monitor elements are controlled through the COLLECT REQUEST METRICS clause on service superclasses and the mon\_req\_metrics database configuration parameter at the database level. Monitor elements are only collected for a request if the request is processed by an agent in a service subclass whose parent service superclass has request monitor element collection enabled, or if request monitor element collection is enabled for the entire database. If request monitor element have been disabled at the database level, and for a service superclass, the metrics reported in the DETAILS\_XML document stop increasing (or remain at 0 if request metrics were disabled at database activation time).

The schema for the XML document that is returned in the DETAILS\_XML column is available in the file sqllib/misc/DB2MonCommon.xsd. The top level element is system\_metrics.

### XML schema for system\_metrics monitor element

The system\_metrics monitor element contains all the system metrics reported by the MON\_GET\_SERVICE\_SUBCLASS\_DETAILS and MON\_GET\_WORKLOAD\_DETAILS table functions.

#### system\_metrics

System level metrics.

Table 29. system\_metrics

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
system_metrics	system_level_metrics	1	1	System level metrics.

#### system\_level\_metrics

This type defines the metrics that are part of the system level.



Table 30. system\_level\_metrics attributes

Name	Data Type	Use	Description
release	xs:nonNegativeInteger	required	Represents the DB2 product level this event was captured on.

Table 31. system\_level\_metrics elements

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
WLM_QUEUE_TIME_TOTAL	metric_value_type	1	1	wlm_queue_time_total - Workload manager total queue time
WLM_QUEUE_ASSIGNMENTS_TOTAL	metric_value_type	1	1	wlm_queue_assignments_total - Workload manager total queue assignments
FCM_TQ_RECV_WAIT_TIME	metric_value_type	1	1	fcm_tq_recv_wait_time - FCM table queue received wait time
FCM_MESSAGE_RECV_WAIT_TIME	metric_value_type	1	1	fcm_message_recv_wait_time - FCM message received wait time
FCM_TQ_SEND_WAIT_TIME	metric_value_type	1	1	fcm_tq_send_wait_time - FCM table queue send wait time
FCM_MESSAGE_SEND_WAIT_TIME	metric_value_type	1	1	fcm_message_send_wait_time - FCM message send wait time
AGENT_WAIT_TIME	metric_value_type	1	1	agent_wait_time - Agent wait time
AGENT_WAITS_TOTAL	metric_value_type	1	1	agent_waits_total - Total agent waits
LOCK_WAIT_TIME	metric_value_type	1	1	lock_wait_time - Time waited on locks
LOCK_WAITS	metric_value_type	1	1	lock_waits - Lock waits
DIRECT_READ_TIME	metric_value_type	1	1	direct_read_time - Direct read time
DIRECT_READ_REQS	metric_value_type	1	1	direct_read_reqs - Direct read requests
DIRECT_WRITE_TIME	metric_value_type	1	1	direct_write_time - Direct write time
DIRECT_WRITE_REQS	metric_value_type	1	1	direct_write_reqs - Direct write requests
LOG_BUFFER_WAIT_TIME	metric_value_type	1	1	log_buffer_wait_time - Log buffer wait time
NUM_LOG_BUFFER_FULL	metric_value_type	1	1	num_log_buffer_full - Number of full log buffers
LOG_DISK_WAIT_TIME	metric_value_type	1	1	log_disk_wait_time - Log disk wait time
LOG_DISK_WAITS_TOTAL	metric_value_type	1	1	log_disk_waits_total - Total log disk waits
TCPIP_RECV_WAIT_TIME	metric_value_type	1	1	tcpip_recv_wait_time - TCP/IP received wait time
TCPIP_RECVS_TOTAL	metric_value_type	1	1	tcpip_recvs_total - TCP/IP receives total
CLIENT_IDLE_WAIT_TIME	metric_value_type	1	1	client_idle_wait_time - Client idle wait time
IPC_RECV_WAIT_TIME	metric_value_type	1	1	ipc_recv_wait_time - Interprocess communication received wait time

Table 31. *system\_level\_metrics* elements (continued)

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
IPC_RECVS_TOTAL	metric_value_type	1	1	ipc_recv_total - Interprocess communication receives total
IPC_SEND_WAIT_TIME	metric_value_type	1	1	ipc_send_wait_time - Interprocess communication send wait time
IPC_SENDS_TOTAL	metric_value_type	1	1	ipc_sends_total - Interprocess communication send total
TCPIP_SEND_WAIT_TIME	metric_value_type	1	1	tcpip_send_wait_time - TCP/IP send wait time
TCPIP_SENDS_TOTAL	metric_value_type	1	1	tcpip_sends_total - TCP/IP sends total
POOL_WRITE_TIME	metric_value_type	1	1	pool_write_time - Total buffer pool physical write time
POOL_READ_TIME	metric_value_type	1	1	pool_read_time - Total buffer pool physical read time
AUDIT_FILE_WRITE_WAIT_TIME	metric_value_type	1	1	audit_file_write_wait_time - Audit file write wait time
AUDIT_FILE_WRITES_TOTAL	metric_value_type	1	1	audit_file_writes_total - Total audit files written
AUDIT_SUBSYSTEM_WAIT_TIME	metric_value_type	1	1	audit_subsystem_wait_time - Audit subsystem wait time
AUDIT_SUBSYSTEM_WAITS_TOTAL	metric_value_type	1	1	audit_subsystem_waits_total - Total audit subsystem waits
DIAGLOG_WRITE_WAIT_TIME	metric_value_type	1	1	diaglog_write_wait_time - Diagnostic log file write wait time
DIAGLOG_WRITES_TOTAL	metric_value_type	1	1	diaglog_writes_total - Total diagnostic log file writes
FCM_SEND_WAIT_TIME	metric_value_type	1	1	fcm_send_wait_time - FCM send wait time
FCM_RECV_WAIT_TIME	metric_value_type	1	1	fcm_recv_wait_time - FCM received wait time
TOTAL_WAIT_TIME	metric_value_type	1	1	total_wait_time - Total wait time
TOTAL_RQST_TIME	metric_value_type	1	1	total_rqst_time - Total request time
RQSTS_COMPLETED_TOTAL	metric_value_type	1	1	rqsts_completed_total - Total requests completed
TOTAL_APP_RQST_TIME	metric_value_type	1	1	total_app_rqst_time - Total application request time
APP_RQSTS_COMPLETED_TOTAL	metric_value_type	1	1	app_rqsts_completed_total - Total application requests completed
TOTAL_SECTION_SORT_PROC_TIME	metric_value_type	1	1	total_section_sort_proc_time - Total section sort processing time
TOTAL_SECTION_SORT_TIME	metric_value_type	1	1	total_section_sort_time - Total section sort time
TOTAL_SECTION_SORTS	metric_value_type	1	1	total_section_sorts - Total section sorts
ROWS_READ	metric_value_type	1	1	rows_read - Rows read
ROWS_MODIFIED	metric_value_type	1	1	rows_modified - Rows modified
POOL_DATA_L_READS	metric_value_type	1	1	pool_data_l_reads - Buffer pool data logical reads

Table 31. system\_level\_metrics elements (continued)

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
POOL_INDEX_L_READS	metric_value_type	1	1	pool_index_l_reads - Buffer pool index logical reads
POOL_TEMP_DATA_L_READS	metric_value_type	1	1	pool_temp_data_l_reads - Buffer pool temporary data logical reads
POOL_TEMP_INDEX_L_READS	metric_value_type	1	1	pool_temp_index_l_reads - Buffer pool temporary index logical reads
POOL_XDA_L_READS	metric_value_type	1	1	pool_xda_l_reads - Buffer pool XDA data logical reads
POOL_TEMP_XDA_L_READS	metric_value_type	1	1	pool_temp_xda_l_reads - Buffer pool temporary XDA data logical reads
TOTAL_CPU_TIME	metric_value_type	1	1	total_cpu_time - Total CPU time
ACT_COMPLETED_TOTAL	metric_value_type	1	1	act_completed_total - Total completed activities
POOL_DATA_P_READS	metric_value_type	1	1	pool_data_p_reads - Buffer pool data physical reads
POOL_TEMP_DATA_P_READS	metric_value_type	1	1	pool_temp_data_p_reads - Buffer pool temporary data physical reads
POOL_XDA_P_READS	metric_value_type	1	1	pool_xda_p_reads - Buffer pool XDA data physical reads
POOL_TEMP_XDA_P_READS	metric_value_type	1	1	pool_temp_xda_p_reads - Buffer pool temporary XDA data physical reads
POOL_INDEX_P_READS	metric_value_type	1	1	pool_index_p_reads - Buffer pool index physical reads
POOL_TEMP_INDEX_P_READS	metric_value_type	1	1	pool_temp_index_p_reads - Buffer pool temporary index physical reads
POOL_DATA_WRITES	metric_value_type	1	1	pool_data_writes - Buffer pool data writes
POOL_XDA_WRITES	metric_value_type	1	1	pool_xda_writes - Buffer pool XDA data writes
POOL_INDEX_WRITES	metric_value_type	1	1	pool_index_writes - Buffer pool index writes
DIRECT_READS	metric_value_type	1	1	direct_reads - Direct reads from database
DIRECT_WRITES	metric_value_type	1	1	direct_writes - Direct writes to database
ROWS_RETURNED	metric_value_type	1	1	rows_returned - Rows returned
DEADLOCKS	metric_value_type	1	1	deadlocks - Deadlocks detected
LOCK_TIMEOUTS	metric_value_type	1	1	lock_timeouts - Number of lock timeouts
LOCK_ESCALS	metric_value_type	1	1	lock_escalations - Number of lock escalations
FCM_SENDS_TOTAL	metric_value_type	1	1	fcm_sends_total - FCM sends total
FCM_RECVS_TOTAL	metric_value_type	1	1	fcm_recvs_total - FCM receives total
FCM_SEND_VOLUME	metric_value_type	1	1	fcm_send_volume - FCM send volume

Table 31. *system\_level\_metrics* elements (continued)

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
FCM_RECV_VOLUME	metric_value_type	1	1	fcm_recv_volume - FCM received volume
FCM_MESSAGE_SENDS_TOTAL	metric_value_type	1	1	fcm_message_sends_total - Total FCM message sends
FCM_MESSAGE_RECVS_TOTAL	metric_value_type	1	1	fcm_message_recvs_total - Total FCM message receives
FCM_MESSAGE_SEND_VOLUME	metric_value_type	1	1	fcm_message_send_volume - FCM message send volume
FCM_MESSAGE_RECV_VOLUME	metric_value_type	1	1	fcm_message_recv_volume - FCM message received volume
FCM_TQ_SENDS_TOTAL	metric_value_type	1	1	fcm_tq_sends_total - FCM table queue send total
FCM_TQ_RECVS_TOTAL	metric_value_type	1	1	fcm_tq_recvs_total - FCM table queue receives total
FCM_TQ_SEND_VOLUME	metric_value_type	1	1	fcm_tq_send_volume - FCM table queue send volume
FCM_TQ_RECV_VOLUME	metric_value_type	1	1	fcm_tq_recv_volume - FCM table queue received volume
TQ_TOT_SEND_SPILLS	metric_value_type	1	1	tq_tot_send_spills - Total number of table queue buffers overflowed
TCPIP_SEND_VOLUME	metric_value_type	1	1	tcpip_send_volume - TCP/IP send volume
TCPIP_RECV_VOLUME	metric_value_type	1	1	tcpip_recv_volume - TCP/IP received volume
IPC_SEND_VOLUME	metric_value_type	1	1	ipc_send_volume - Interprocess communication send volume
IPC_RECV_VOLUME	metric_value_type	1	1	ipc_recv_volume - Interprocess communication received volume
POST_THRESHOLD_SORTS	metric_value_type	1	1	post_threshold_sorts - Post threshold sorts
POST_SHRTHRESHOLD_SORTS	metric_value_type	1	1	post_shrthreshold_sorts - Post shared threshold sorts
SORT_OVERFLOWS	metric_value_type	1	1	sort_overflows - Sort overflows
AUDIT_EVENTS_TOTAL	metric_value_type	1	1	audit_events_total - Total audit events
TOTAL_RQST_MAPPED_IN	metric_value_type	0	1	total_rqst_mapped_in - Total request mapped-in
TOTAL_RQST_MAPPED_OUT	metric_value_type	0	1	total_rqst_mapped_out - Total request mapped-out
ACT_REJECTED_TOTAL	metric_value_type	1	1	act_rejected_total - Total rejected activities
ACT_ABORTED_TOTAL	metric_value_type	1	1	act_aborted_total - Total aborted activities
TOTAL_SORTS	metric_value_type	1	1	total_sorts - Total sorts

### **metric\_value\_type**

A type for the value of metric times and counters.

## Capturing activity monitor elements using the activity event monitor

The activity event monitor contains the `details_xml` monitor element in the `event_activity` logical data groups. Use this monitor element to capture information about activities.

The monitor element **`details_xml`** returns an XML document containing all the activity monitor elements reported by the `MON_GET_ACTIVITY_DETAILS` table function. Activity monitor elements are a subset of the activity details document reported in the `DETAILS` column of the `MON_GET_ACTIVITY_DETAILS` table function.

Activity monitor elements are controlled through the `COLLECT ACTIVITY METRICS` clause on workloads, or the **`mon_act_metrics`** database configuration parameter at the database level. Monitor elements are collected if the connection that submits the activity is associated with a workload or database for which activity monitor element collection is enabled. If activity monitor element are not collected for an activity, `DETAILS_XML` contains an empty document.

The schema for the XML document that is returned in the `DETAILS_XML` column is available in the file `sqllib/misc/DB2MonCommon.xsd`. The top level element is `activity_metrics`.

### XML schema for activity\_metrics monitor element

The `activity_metrics` monitor element contains all the activity metrics reported by the `MON_GET_ACTIVITY_DETAILS` table function.

#### activity\_metrics

Activity level metrics.

Table 32. *activity\_metrics*

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
<code>activity_metrics</code>	<code>activity_level_metrics</code>	1	1	Activity level metrics.

#### activity\_level\_metrics

This type defines the metrics that are part of the activity level.

Table 33. *activity\_level\_metrics attributes*

Name	Data Type	Use	Description
<code>release</code>	<code>xs:nonNegativeInteger</code>	required	Represents the DB2 product level this event was captured on.

Table 34. *activity\_level\_metrics elements*

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
<code>WLM_QUEUE_TIME_TOTAL</code>	<code>metric_value_type</code>	1	1	<code>wlm_queue_time_total</code> - Workload manager total queue time

Table 34. activity\_level\_metrics elements (continued)

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
WLM_QUEUE_ASSIGNMENTS_TOTAL	metric_value_type	1	1	wlm_queue_assignments_total - Workload manager total queue assignments
FCM_TQ_RECV_WAIT_TIME	metric_value_type	1	1	fcm_tq_recv_wait_time - FCM table queue received wait time
FCM_MESSAGE_RECV_WAIT_TIME	metric_value_type	1	1	fcm_message_recv_wait_time - FCM message received wait time
FCM_TQ_SEND_WAIT_TIME	metric_value_type	1	1	fcm_tq_send_wait_time - FCM table queue send wait time
FCM_MESSAGE_SEND_WAIT_TIME	metric_value_type	1	1	fcm_message_send_wait_time - FCM message send wait time
LOCK_WAIT_TIME	metric_value_type	1	1	lock_wait_time - Time waited on locks
LOCK_WAITS	metric_value_type	1	1	lock_waits - Lock waits
DIRECT_READ_TIME	metric_value_type	1	1	direct_read_time - Direct read time
DIRECT_READ_REQS	metric_value_type	1	1	direct_read_reqs - Direct read requests
DIRECT_WRITE_TIME	metric_value_type	1	1	direct_write_time - Direct write time
DIRECT_WRITE_REQS	metric_value_type	1	1	direct_write_reqs - Direct write requests
LOG_BUFFER_WAIT_TIME	metric_value_type	1	1	log_buffer_wait_time - Log buffer wait time
NUM_LOG_BUFFER_FULL	metric_value_type	1	1	num_log_buffer_full - Number of full log buffers
LOG_DISK_WAIT_TIME	metric_value_type	1	1	log_disk_wait_time - Log disk wait time
LOG_DISK_WAITS_TOTAL	metric_value_type	1	1	log_disk_waits_total - Total log disk waits
POOL_WRITE_TIME	metric_value_type	1	1	pool_write_time - Total buffer pool physical write time
POOL_READ_TIME	metric_value_type	1	1	pool_read_time - Total buffer pool physical read time
AUDIT_FILE_WRITE_WAIT_TIME	metric_value_type	1	1	audit_file_write_wait_time - Audit file write wait time
AUDIT_FILE_WRITES_TOTAL	metric_value_type	1	1	audit_file_writes_total - Total audit files written
AUDIT_SUBSYSTEM_WAIT_TIME	metric_value_type	1	1	audit_subsystem_wait_time - Audit subsystem wait time
AUDIT_SUBSYSTEM_WAITS_TOTAL	metric_value_type	1	1	audit_subsystem_waits_total - Total audit subsystem waits
DIAGLOG_WRITE_WAIT_TIME	metric_value_type	1	1	diaglog_write_wait_time - Diagnostic log file write wait time
DIAGLOG_WRITES_TOTAL	metric_value_type	1	1	diaglog_writes_total - Total diagnostic log file writes
FCM_SEND_WAIT_TIME	metric_value_type	1	1	fcm_send_wait_time - FCM send wait time
FCM_RECV_WAIT_TIME	metric_value_type	1	1	fcm_recv_wait_time - FCM received wait time

Table 34. activity\_level\_metrics elements (continued)

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
TOTAL_ACT_WAIT_TIME	metric_value_type	1	1	total_act_wait_time - Total activity wait time
TOTAL_SECTION_SORT_PROC_TIME	metric_value_type	1	1	total_section_sort_proc_time - Total section sort processing time
TOTAL_SECTION_SORT_TIME	metric_value_type	1	1	total_section_sort_time - Total section sort time
TOTAL_SECTION_SORTS	metric_value_type	1	1	total_section_sorts - Total section sorts
TOTAL_ACT_TIME	metric_value_type	1	1	total_act_time - Total activity time
ROWS_READ	metric_value_type	1	1	rows_read - Rows read
ROWS_MODIFIED	metric_value_type	1	1	rows_modified - Rows modified
POOL_DATA_L_READS	metric_value_type	1	1	pool_data_l_reads - Buffer pool data logical reads
POOL_INDEX_L_READS	metric_value_type	1	1	pool_index_l_reads - Buffer pool index logical reads
POOL_TEMP_DATA_L_READS	metric_value_type	1	1	pool_temp_data_l_reads - Buffer pool temporary data logical reads
POOL_TEMP_INDEX_L_READS	metric_value_type	1	1	pool_temp_index_l_reads - Buffer pool temporary index logical reads
POOL_XDA_L_READS	metric_value_type	1	1	pool_xda_l_reads - Buffer pool XDA data logical reads
POOL_TEMP_XDA_L_READS	metric_value_type	1	1	pool_temp_xda_l_reads - Buffer pool temporary XDA data logical reads
TOTAL_CPU_TIME	metric_value_type	1	1	total_cpu_time - Total CPU time
POOL_DATA_P_READS	metric_value_type	1	1	pool_data_p_reads - Buffer pool data physical reads
POOL_TEMP_DATA_P_READS	metric_value_type	1	1	pool_temp_data_p_reads - Buffer pool temporary data physical reads
POOL_XDA_P_READS	metric_value_type	1	1	pool_xda_p_reads - Buffer pool XDA data physical reads
POOL_TEMP_XDA_P_READS	metric_value_type	1	1	pool_temp_xda_p_reads - Buffer pool temporary XDA data physical reads
POOL_INDEX_P_READS	metric_value_type	1	1	pool_index_p_reads - Buffer pool index physical reads
POOL_TEMP_INDEX_P_READS	metric_value_type	1	1	pool_temp_index_p_reads - Buffer pool temporary index physical reads
POOL_DATA_WRITES	metric_value_type	1	1	pool_data_writes - Buffer pool data writes
POOL_XDA_WRITES	metric_value_type	1	1	pool_xda_writes - Buffer pool XDA data writes
POOL_INDEX_WRITES	metric_value_type	1	1	pool_index_writes - Buffer pool index writes
DIRECT_READS	metric_value_type	1	1	direct_reads - Direct reads from database

Table 34. activity\_level\_metrics elements (continued)

Name	Data Type	Minimum Occurrence	Maximum Occurrence	Description
DIRECT_WRITES	metric_value_type	1	1	direct_writes - Direct writes to database
ROWS_RETURNED	metric_value_type	1	1	rows_returned - Rows returned
DEADLOCKS	metric_value_type	1	1	deadlocks - Deadlocks detected
LOCK_TIMEOUTS	metric_value_type	1	1	lock_timeouts - Number of lock timeouts
LOCK_ESCALS	metric_value_type	1	1	lock_escalations - Number of lock escalations
FCM_SENDS_TOTAL	metric_value_type	1	1	fcm_sends_total - FCM sends total
FCM_RECVS_TOTAL	metric_value_type	1	1	fcm_recvs_total - FCM receives total
FCM_SEND_VOLUME	metric_value_type	1	1	fcm_send_volume - FCM send volume
FCM_RECV_VOLUME	metric_value_type	1	1	fcm_recv_volume - FCM received volume
FCM_MESSAGE_SENDS_TOTAL	metric_value_type	1	1	fcm_message_sends_total - Total FCM message sends
FCM_MESSAGE_RECVS_TOTAL	metric_value_type	1	1	fcm_message_recvs_total - Total FCM message receives
FCM_MESSAGE_SEND_VOLUME	metric_value_type	1	1	fcm_message_send_volume - FCM message send volume
FCM_MESSAGE_RECV_VOLUME	metric_value_type	1	1	fcm_message_recv_volume - FCM message received volume
FCM_TQ_SENDS_TOTAL	metric_value_type	1	1	fcm_tq_sends_total - FCM table queue send total
FCM_TQ_RECVS_TOTAL	metric_value_type	1	1	fcm_tq_recvs_total - FCM table queue receives total
FCM_TQ_SEND_VOLUME	metric_value_type	1	1	fcm_tq_send_volume - FCM table queue send volume
FCM_TQ_RECV_VOLUME	metric_value_type	1	1	fcm_tq_recv_volume - FCM table queue received volume
TQ_TOT_SEND_SPILLS	metric_value_type	1	1	tq_tot_send_spills - Total number of table queue buffers overflowed
POST_THRESHOLD_SORTS	metric_value_type	1	1	post_threshold_sorts - Post threshold sorts
POST_SHRTHRESHOLD_SORTS	metric_value_type	1	1	post_shrthreshold_sorts - Post shared threshold sorts
SORT_OVERFLOWS	metric_value_type	1	1	sort_overflows - Sort overflows
AUDIT_EVENTS_TOTAL	metric_value_type	1	1	audit_events_total - Total audit events
TOTAL_SORTS	metric_value_type	1	1	total_sorts - Total sorts

### **metric\_value\_type**

A type for the value of metric times and counters.



---

## Event monitors that write to tables, files, and pipes

Some event monitors can be configured to write information about database events to tables, pipes, or files.

Event monitors are used to collect information about the database and any connected applications when specified events occur. Events represent transitions in database activity such as connections, deadlocks, statements, or transactions. You can define an event monitor by the type of event or events you want it to monitor. For example, a deadlock event monitor waits for a deadlock to occur; when one does, it collects information about the applications involved and the locks in contention.

To create an event monitor, use the `CREATE EVENT MONITOR` SQL statement. Event monitors collect event data only when they are active. To activate or deactivate an event monitor, use the `SET EVENT MONITOR STATE` SQL statement. The status of an event monitor (whether it is active or inactive) can be determined by the SQL function `EVENT_MON_STATE`.

When the `CREATE EVENT MONITOR` SQL statement is executed, the definition of the event monitor it creates is stored in the following database system catalog tables:

- `SYSCAT.EVENTMONITORS`: event monitors defined for the database.
- `SYSCAT.EVENTS`: events monitored for the database.
- `SYSCAT.EVENTTABLES`: target tables for table event monitors.

Each event monitor has its own private logical view of the instance's data in the monitor elements. If a particular event monitor is deactivated and then reactivated, its view of these counters is reset. Only the newly activated event monitor is affected; all other event monitors will continue to use their view of the counter values (plus any new additions).

Event monitor output can be directed to non-partitioned SQL tables, a file, or a named pipe.

**Note:** The deprecated detailed deadlock event monitor, `DB2DETAILDEADLOCK`, is created by default for each database and starts when the database is activated. Avoid the overhead this event monitor incurs by dropping it. The use of the `DB2DETAILDEADLOCK` monitor element is no longer recommended. This deprecated event monitor might be removed in a future release. Use the `CREATE EVENT MONITOR FOR LOCKING` statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

## Collecting information about database system events

Event monitors are used to collect information about the database and any connected applications when specified events occur. Event monitors are database objects, and as such, they are created and manipulated using SQL data definition language (SQL DDL) statements.

You will need `SQLADM` or `DBADM` authority to create and manipulate event monitors.

A certain type of event monitor gives you the choice to direct the output to a file, pipe, or (regular) table. This type of monitor is described here. Some of these details do not apply to event monitors that direct output to an unformatted event table.

The steps listed below represent a typical life cycle of an event monitor. These steps need not necessarily be executed in the presented order, if at all. For instance, depending on usage it is possible that an event monitor is never dropped or even deactivated. There are, however, two constants in an event monitor's life cycle: the first step will always be the creation of the event monitor and the last step will always be the deletion of the event monitor.

1. "Creating an event monitor" on page 60.

2. *For file and pipe event monitors only:*

- Ensure that the directory or named pipe that will receive the event records exists. The event monitor will not activate otherwise.

On AIX®, you can create named pipes by using the `mkfifo` command. On Linux® and other UNIX® types (such as the Solaris operating system) use the `pipe()` routine.

On Windows®, you can create named pipes by using the `CreateNamedPipe()` routine.

- *For pipe event monitors only:* Open the named pipe prior to activating the event monitor. This can be done with an operating system function:

– for UNIX: `open()`

– for Windows: `ConnectNamedPipe()`

This can also be done with the `db2evmon` executable:

```
db2evmon -db databasename
          -evm eventmonname
```

*databasename* represents the name of the database being monitored.

*eventmonname* represents the name of the event monitor.

3. Activate the newly created event monitor to enable it to collect information.

```
SET EVENT MONITOR eventmonname STATE 1;
```

If the event monitor was created with the `AUTOSTART` option, the event monitor will be activated when the first user connects to the database. Furthermore, once an event monitor has been explicitly activated, it will automatically restart whenever the database is reactivated. Restarting occurs until the event monitor is explicitly deactivated or until the instance is stopped. When a table event monitor is started, the event monitor updates the `evmon_activates` column of the `SYSCAT.EVENTMONITORS` catalog table. This change is logged, so the `DATABASE CONFIGURATION` will display:

```
Database is consistent = NO
```

If an event monitor is created with the `AUTOSTART` option, and the first user connects to the database and immediately disconnects so that the database is deactivated, a log file will be produced.

4. To see if an event monitor is active or inactive, issue the SQL function `EVENT_MON_STATE` in a query against the table, `SYSCAT.EVENTMONITORS`:

```
SELECT eventmonname, EVENT_MON_STATE(eventmonname) FROM
       syscat.eventmonitors;
```

A list of all existing event monitors will be listed, along with their status. A returned value of 0 indicates that the specified event monitor is inactive, and 1 indicates that it is active.

5. Read event monitor output. For write-to-table event monitors this involves examining the target tables. To access file or pipe event monitor data from the CLP see the Related task, "Formatting file or pipe event monitor output from a command line".
6. To deactivate, or turn off an event monitor, use the SET EVENT MONITOR statement:

```
SET EVENT MONITOR evmonname STATE 0
```

Deactivating an event monitor does not result in its deletion. It will exist as a dormant database object. Deactivating an event monitor will flush all its contents. Hence, if you reactivate a deactivated event monitor, it will only contain information collected since its reactivation.

Note that if an activity event monitor is active when the database deactivates, any backlogged activity records in the queue are discarded. To ensure that you obtain all activity event monitor records and that none are discarded, explicitly deactivate the activity event monitor first before deactivating the database.

When an activity event monitor is explicitly deactivated, all backlogged activity records in the queue are processed before the event monitor deactivates.

7. After you deactivate a pipe event monitor, close the corresponding named pipe. In UNIX use the `close()` function, and in Windows 2000 use the `DisconnectNamedPipe()` function.
8. To eliminate an event monitor object, use the DROP EVENT MONITOR statement:

```
DROP EVENT MONITOR evmonname
```

You can only drop an event monitor if it is inactive.

9. After you drop a pipe event monitor, delete the corresponding named pipe. In UNIX use the `unlink()` function, and in Windows use the `CloseHandle()` function. When dropping a write-to-table event monitor, the associated target tables are not dropped. Similarly, when dropping a file event monitor, the associated files are not deleted.

## Creating an event monitor

The first step in an event monitor's life cycle is its creation. Before you create an event monitor, you must determine where the event records are to be sent: to SQL tables, files, or through named pipes.

You will need SQLADM or DBADM authority to create an event monitor.

For each event record destination there are particular options that are to be specified in the CREATE EVENT MONITOR SQL statement. The target table of a CREATE EVENT MONITOR statement must be a non-partitioned table. Monitoring events in a partitioned database also requires special attention.

A certain type of event monitor gives you the choice to direct the output to a file, pipe, or (regular) table. This type of monitor is described here. Some of these details do not apply to event monitors that direct output to an unformatted event table.

1. Create a table event monitor.
2. Create a file event monitor.

3. Create a pipe event monitor.
4. Create an event monitor for a partitioned database.

Once an event monitor is created and activated, it will record monitoring data as its specified events occur.

### Creating a table event monitor

When creating an event monitor you must determine where the information it collects is to be stored. A table event monitor streams event records to SQL tables, presenting a simple alternative to file and pipe event monitors in enabling you to easily capture, parse, and manage event monitoring data. For every event type an event monitor collects, target tables are created for each of the associated logical data groups.

You will need SQLADM or DBADM authority to create a table event monitor.

The target table of a CREATE EVENT MONITOR statement must be a non-partitioned table.

The various options for table event monitors are set in the CREATE EVENT MONITOR statement. For further assistance in generating CREATE EVENT MONITOR SQL statements for write-to-table event monitors, you can use the db2evtbl command. Simply provide the name of the event monitor and the desired event type (or types), and the CREATE EVENT MONITOR statement is generated, complete with listings of all the target tables. You can then copy the generated statement, make modifications, and then execute the statement from the CLP.

1. Indicate that event monitor data is to be collected in a table (or set of tables).

```
CREATE EVENT MONITOR dlmon FOR eventtype
      WRITE TO TABLE
```

dlmon is the name of the event monitor.

2. Specify the types of events to be monitored. You can monitor one or more event types with a single event monitor.

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
      WRITE TO TABLE
```

This event monitor will monitor for the CONNECTIONS and DEADLOCKS WITH DETAILS event types. Assuming that the above statement was issued by the user 'riihi', the derived names and table spaces of the target tables are as follows:

- riihi.connheader\_dlmon
- riihi.conn\_dlmon
- riihi.connmemuse\_dlmon
- riihi.deadlock\_dlmon
- riihi.dlconn\_dlmon
- riihi.dllock\_dlmon
- riihi.control\_dlmon

3. Specify the size of the table event monitor buffers (in 4K pages) by adjusting the BUFFERSIZE value:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
      WRITE TO TABLE BUFFERSIZE 8
```

8 is the combined capacity (in 4K pages) of the two event table buffers. This adds up to 32K of buffer space; 16K for each buffer.

The default size of each buffer is 4 pages (two 16K buffers are allocated). The minimum size is 1 page. The maximum size of the buffers is limited by the size of the monitor heap, because the buffers are allocated from that heap. For performance reasons, highly active event monitors should have larger buffers than relatively inactive event monitors.

4. Indicate if you need the event monitor to be blocked or non-blocked. For blocked event monitors, each agent that generates an event will wait for the event buffers to be written to table if they are full. This can degrade database performance, as the suspended agent and any dependent agents cannot run until the buffers are clear. Use the `BLOCKED` clause to ensure no losses of event data:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
WRITE TO TABLE BUFFERSIZE 8 BLOCKED
```

Event monitors are blocked by default.

If database performance is of greater importance than collecting every single event record, use non-blocked event monitors. In this case, each agent that generates an event will not wait for the event buffers to be written to table if they are full. As a result, non-blocked event monitors are subject to data loss on highly active systems. Use the `NONBLOCKED` clause to minimize the performance overhead from event monitoring:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
WRITE TO TABLE BUFFERSIZE 8 NONBLOCKED
```

5. Indicate the logical data groups from which you need to collect event records. Event monitors store the data from each logical data group in corresponding tables.

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
WRITE TO TABLE CONN, DLCONN, DLLOCK
BUFFERSIZE 8 NONBLOCKED
```

The `CONN`, `DLCONN`, and `DLLOCK` logical data groups are selected. Not mentioned are the other available logical data groups, `CONNHEADER`, `DEADLOCK`, or `CONTROL`. Data relevant to `CONNHEADER`, `DEADLOCK`, or `CONTROL` will not be stored for the `dlmon` event monitor.

6. Indicate the monitor elements for which you need to collect data.

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
WRITE TO TABLE CONN,
DLCONN (EXCLUDES(agent_id, lock_wait_start_time)),
DLLOCK (INCLUDES(lock_mode, table_name))
BUFFERSIZE 8 NONBLOCKED
```

All the monitor elements for `CONN` are captured (this is the default behavior). For `DLCONN`, all monitor elements except `agent_id` and `lock_wait_start_time` are captured. And finally, for `DLLOCK`, `lock_mode` and `table_name` are the only monitor elements captured.

7. Provide names for the tables to be created, and designate a table space:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
WRITE TO TABLE CONN,
DLCONN (TABLE mydept.dlconnections
EXCLUDES(agent_id, lock_wait_start_time)),
DLLOCK (TABLE dllocks IN mytablespace
INCLUDES(lock_mode, table_name))
BUFFERSIZE 8 NONBLOCKED
```

Assuming that the above statement was issued by the user `riihi`, the derived names and table spaces of the target tables are as follows:

- `CONN`: `riihi.conn_dlmon` (on the default table space)
- `DLCONN`: `mydept.dlconnections` (on the default table space)
- `DLLOCK`: `riihi.dllocks` (on the `MYTABLESPACE` table space)

The default table space is assigned from `IBMDEFAULTGROUP`, provided the event monitor definer has `USE` privileges. If the definer does not have `USE` privileges over this table space, then a table space over which the definer does have `USE` privileges will be assigned.

8. Indicate how full the table space can get before the event monitor automatically deactivates:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                        WRITE TO TABLE DLCONN PCTDEACTIVATE 90
                        BUFFERSIZE 8 NONBLOCKED
```

When the table space reaches 90% capacity, the `dlmon` event monitor automatically shuts off. The `PCTDEACTIVATE` clause can only be used for DMS table spaces. If the target table space has auto-resize enabled, set the `PCTDEACTIVATE` clause to 100.

9. Specify if the event monitor is to be activated automatically each time the database starts. By default, event monitors (with the exception of the WLM event monitors) are not activated automatically when the database starts.

- To create an event monitor that starts automatically when the database starts, issue the following statement:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                        WRITE TO TABLE BUFFERSIZE 8 NONBLOCKED
                        AUTOSTART NONBLOCKED
```

- To create an event monitor that does not start automatically when the database starts, issue the following statement:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                        WRITE TO TABLE BUFFERSIZE 8 NONBLOCKED
                        MANUALSTART
```

10. To activate or deactivate an event monitor, use the `SET EVENT MONITOR STATE` statement.

Once a table event monitor is created and activated, it will record monitoring data as its specified events occur.

## Event monitor table management

You can define an event monitor to store its event records in SQL tables. To do this, use the `CREATE EVENT MONITOR` statement with the `WRITE TO TABLE` clause.

Upon the creation of a write-to-table event monitor, the database creates *target tables* to store records for each of the logical data groups returning data. By default, the database creates the tables in the event monitor creator's schema, and names the tables according to their corresponding logical data group and event monitor name. In each table, the column names match the monitor element names that they represent.

For example, the user `riihi` is creating an event monitor that captures `STATEMENTS` events:

```
CREATE EVENT MONITOR foo FOR STATEMENTS WRITE TO TABLE
```

Event monitors using the STATEMENTS event type collect data from the event\_connheader, event\_stmt, and event\_subsection logical data groups. The database created the following tables:

- riihi.connheader\_foo
- riihi.stmt\_foo
- riihi.subsection\_foo
- riihi.control\_foo

In addition to the tables representing logical data groups specific to individual event types, a control table is created for every write-to-table event monitor. This is represented above by riihi.control\_foo. A control table contains event monitor metadata, specifically, from the event\_start, event\_db\_header (**conn\_time** monitor element only), and event\_overflow logical data groups.

Whenever a write-to-table event monitor activates, it will acquire IN table locks on each target table in order to prevent them from being modified while the event monitor is active. Table locks are maintained on all tables while the event monitor is active. If exclusive access is required on any of the target tables (for example, when a utility is to be run), first deactivate the event monitor to release the table locks before attempting such access.

Each column name in a target table matches an event monitor element identifier. Any event monitor element that does not have a corresponding target table column is ignored.

Write-to-table event monitor target tables, including the unformatted event (UE) table, must be pruned manually. On highly active systems, event monitors can quickly fill machine space due to the high volume of data they record. Unlike event monitors that write to files or named pipes, you can define write-to-table event monitors to record only certain logical data groups, or monitor elements. This feature enables you to collect only the data relevant to your purposes and reduce the volume of data generated by the event monitors. For example, the following statement defines an event monitor that captures TRANSACTIONS events, but only from the event\_xact logical data group, and including only the **lock\_escal** monitor element:

```
CREATE EVENT MONITOR foo_lite FOR TRANSACTIONS WRITE TO TABLE
      XACT(INCLUDES(lock_escal))
```

**Note:** This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR UNIT OF WORK statement to monitor transaction events.

There are circumstances where it may not be desirable to have the event monitor's target tables residing in the default schema, with default table names, in the default table space. For instance, you may want the target tables to exist in their own table space if you are anticipating high volumes of monitoring data.

You can specify the schema, table, and table space names in the CREATE EVENT MONITOR statement. The schema name is provided along with the table name, forming a derived name for the table.

A target table can only be used by a single event monitor. If a target table is found to already be defined for another event monitor, or if it cannot be created for any other reason, the CREATE EVENT MONITOR statement will fail.



The table space name can be added after the table name with the optional IN clause. Unlike the target tables, which DB2 automatically creates, a table space must already exist if it is included in an event monitor definition. If no table space is specified, then a table space over which the definer has USE privileges will be assigned.

In a partitioned database environment, a write-to-table event monitor will only be active on database partitions where the table space containing the event monitor table exists. When the target table space for an active event monitor does not exist on a particular database partition, the event monitor will be deactivated on that database partition, and an error is written to the db2diag log file.

For increased performance in retrieving event monitor data, you can create indexes for the event tables. You can also add additional table attributes, such as triggers, relational integrity, and constraints. The event monitor will ignore them.

For example, the following statement defines an event monitor that captures STATEMENTS events, using the event\_connheader, event\_stmt, and event\_subsection logical data groups. Each of the three target tables has different schema, table and table space combinations:

```
CREATE EVENT MONITOR foo FOR STATEMENTS
WRITE TO TABLE CONNHEADER,
STMT (TABLE mydept.statements),
SUBSECTION (TABLE subsections, IN mytablespace)
```

Assuming that the above statement was issued by the user 'riihi', the derived names and table spaces of the target tables are as follows:

- CONNHEADER: riihi.connheader\_foo (on the default table space)
- STMT: mydept.statements (on the default table space)
- SUBSECTION: riihi.subsections (on the MYTABLESPACE table space)

If a target table does not exist when the event monitor activates, activation continues and data that would otherwise be inserted into the target table is ignored. Correspondingly, if a monitor element does not have a dedicated column in the target table, it is ignored.

For active write-to-table event monitors there is a risk that the table spaces storing event records can reach their capacity. To control this risk for DMS table spaces you can define at which percentage of table space capacity the event monitor will deactivate. This can be declared in the PCTDEACTIVATE clause in the CREATE EVENT MONITOR statement.

For SMS table spaces, the value is set to 100. It is recommended that when the target table space has auto-resize enabled the PCTDEACTIVATE be set to 100.

In a non-partitioned database environment, all write to table event monitors are deactivated when the last application terminates (and the database has not been explicitly activated). In a partitioned database environment, write to table event monitors are deactivated when the catalog partition deactivates.

The following table presents the default target table names as sorted by the event type for which they are returned.



Table 35. Write-to-Table Event Monitor Target Tables

Event type	Target table names	Available information
DEADLOCKS <sup>1</sup>	CONNHEADER	Connection metadata
	DEADLOCK	Deadlock data
	DLCONN	Applications and locks involved in deadlock
	CONTROL	Event monitor metadata
DEADLOCKS WITH DETAILS <sup>1</sup>	CONNHEADER	Connection metadata
	DEADLOCK	Deadlock data
	DLCONN	Applications involved in deadlock
	DLLOCK	Locks involved in deadlock
	CONTROL	Event monitor metadata
DEADLOCKS WITH DETAILS HISTORY <sup>1</sup>	CONNHEADER	Connection metadata
	DEADLOCK	Deadlock data
	DLCONN	Applications involved in deadlock
	DLLOCK	Locks involved in deadlock
	STMTHIST	List of the previous statements in the unit of work
	CONTROL	Event monitor metadata
DEADLOCKS WITH DETAILS HISTORY VALUES <sup>1</sup>	CONNHEADER	Connection metadata
	DEADLOCK	Deadlock data
	DLCONN	Applications involved in deadlock
	DLLOCK	Locks involved in deadlock
	STMTHIST	List of the previous statements in the unit of work
	STMTVALS	Input Data values of statements in STMTHIST table
	CONTROL	Event monitor metadata
STATEMENT	CONNHEADER	Connection metadata
	STMT	Statement data
	SUBSECTION	Statement data specific to subsection
	CONTROL	Event monitor metadata
TRANSACTIONS <sup>2</sup>	CONNHEADER	Connection metadata
	XACT	Transaction data
	CONTROL	Event monitor metadata
CONNECTIONS	CONNHEADER	Connection metadata
	CONN	Connection data
	CONTROL	Event monitor metadata
	CONNMEMUSE	Memory pool metadata
DATABASE	DB	Database manager data
	CONTROL	Event monitor metadata
	DBMEMUSE	Memory pool metadata
BUFFERPOOLS	BUFFERPOOL	Buffer pool data
	CONTROL	Event monitor metadata
TABLESPACES	TABLESPACE	Tablespace data
	CONTROL	Event monitor metadata

Table 35. Write-to-Table Event Monitor Target Tables (continued)

Event type	Target table names	Available information
TABLES	TABLE	Table data
	CONTROL	Event monitor metadata
ACTIVITIES	ACTIVITY	Activities that completed executing or were captured in progress.
	ACTIVITYSTMT	Statement information for activities that are statements.
	ACTIVITYVALS	Input data values for activities that have them. The data types that can be reported excludes the following: CLOB, REF, BOOLEAN, STRUCT, DATALINK, LONG VARGRAPHIC, LONG, XMLLOB, and DBCLOB.
	CONTROL	Event monitor metadata
STATISTICS	SCSTATS	Statistics computed from the activities that executed within each service class, work class, or workload in the system.
	WCSTATS	
	WLSTATS	
	HISTOGRAMBIN	
	QSTATS	
THRESHOLD VIOLATIONS	THRESHOLDVIOLATIONS	List of thresholds that have been violated as well as the times of violations.
	CONTROL	Event monitor metadata

<sup>1</sup> This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

<sup>2</sup> This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR UNIT OF WORK statement to monitor transaction events.

The following logical data groups are not collected for write-to-table event monitors:

- event\_log\_stream\_header
- event\_log\_header
- event\_dbheader (only the **conn\_time** monitor element is collected)

The data type of each column in an event monitor table corresponds to the data type of the monitor element represented by the column. The following is a set of data type mappings that correspond the original system monitor data types of the monitor elements (found in sqlmon.h) to the SQL data types of the table columns.

Table 36. System Monitor Data Type Mappings

System monitor data type	SQL data type
SQLM_TYPE_STRING	CHAR[n], VARCHAR[n], CLOB[n]
SQLM_TYPE_U8BIT and SQLM_TYPE_8BIT	SMALLINT, INTEGER, or BIGINT
SQLM_TYPE_U16BIT and SQLM_TYPE_16BIT	SMALLINT, INTEGER, or BIGINT
SQLM_TYPE_U32BIT and SQLM_TYPE_32BIT	INTEGER or BIGINT
SQLM_TYPE_U64BIT and SQLM_TYPE_64BIT	BIGINT

Table 36. System Monitor Data Type Mappings (continued)

System monitor data type	SQL data type
SQLM_TIMESTAMP	TIMESTAMP
SQLM_TIME	BIGINT
SQLCA: SQLERRMC	VARCHAR[72]
SQLCA: SQLSTATE	CHAR[5]
SQLCA: SQLWARN	CHAR[11]
SQLCA: other fields	INTEGER or BIGINT
SQLM_TYPE_HANDLE	BLOB[n]

**Note:**

1. All columns are NOT NULL.
2. Because the performance of tables with CLOB columns is inferior to tables that have VARCHAR columns, consider using the TRUNC keyword when specifying the stmt evmGroup (or dlconn evmGroup, when using deadlocks with details).
3. SQLM\_TYPE\_HANDLE is used to represent the compilation environment handle object.

**Creating a file event monitor**

When creating an event monitor you must determine where the information it collects is to be stored. File event monitors store event records in files. File event monitors and their options are defined by the CREATE EVENT MONITOR statement.

You will need SQLADM or DBADM authority to create a file event monitor.

A file event monitor streams event records to a series of 8-character numbered files with the extension "evt" (for example, 00000000.evt, 00000001.evt, and 00000002.evt). The data should be considered to be one logical file even though the data is broken up into smaller pieces (that is, the start of the data stream is the first byte in the file 00000000.evt; the end of the data stream is the last byte in the file nnnnnnnn.evt). An event monitor will never span a single event record across two files.

1. Indicate that event monitor data is to be collected in a file (or set of files), and provide a directory location where event files are to be stored.

```
CREATE EVENT MONITOR dlmon FOR eventtype
        WRITE TO FILE '/tmp/dlevents'
```

dlmon is the name of the event monitor.

/tmp/dlevents is the name of the directory path (on UNIX) where the event monitor is to write the event files.

2. Specify the types of events to be monitored. You can monitor one or more event types with a single event monitor.

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
        WRITE TO FILE '/tmp/dlevents'
```

This event monitor will monitor for the CONNECTIONS and DEADLOCKS WITH DETAILS event types.

3. Specify the size of the file event monitor buffers (in 4K pages) by adjusting the BUFFERSIZE value:

```
CREATE EVENT MONITOR d1mon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
WRITE TO FILE '/tmp/d1events' BUFFERSIZE 8
```

8 is the capacity in 4K pages of the two event file buffers.

The default size of each buffer is 4 pages (two 16K buffers are allocated). The minimum size is 1 page. The maximum size of the buffers is limited by the size of the monitor heap, because the buffers are allocated from that heap. For performance reasons, highly active event monitors should have larger buffers than relatively inactive event monitors.

4. Indicate if you need the event monitor to be blocked or non-blocked. For blocked event monitors, each agent that generates an event will wait for the event buffers to be written to file if they are full. This can degrade database performance, as the suspended agent and any dependent agents cannot run until the buffers are clear. Use the `BLOCKED` clause to ensure no losses of event data:

```
CREATE EVENT MONITOR d1mon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
WRITE TO FILE '/tmp/d1events' BUFFERSIZE 8
BLOCKED
```

Event monitors are blocked by default. If database performance is of greater importance than collecting every single event record, use non-blocked event monitors. In this case, each agent that generates an event will not wait for the event buffers to be written to file if they are full. As a result, non-blocked event monitors are subject to data loss on highly active systems. Use the `NONBLOCKED` clause to minimize the performance overhead from event monitoring:

```
CREATE EVENT MONITOR d1mon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
WRITE TO FILE '/tmp/d1events' BUFFERSIZE 8
NONBLOCKED
```

5. Specify the maximum number of event files that can be collected for an event monitor. If this limit is reached, the event monitor will deactivate itself.

```
CREATE EVENT MONITOR d1mon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
WRITE TO FILE '/tmp/d1events' BUFFERSIZE 8
NONBLOCKED MAXFILES 5
```

5 is the maximum number of event files that will be created.

You can also specify that there is no limit to the number of event files that the event monitor can create:

```
CREATE EVENT MONITOR d1mon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
WRITE TO FILE '/tmp/d1events' BUFFERSIZE 8
NONBLOCKED MAXFILES NONE
```

6. Specify the maximum size (in 4K pages) for each event file created by an event monitor. If this limit is reached, a new file is created.

```
CREATE EVENT MONITOR d1mon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
WRITE TO FILE '/tmp/d1events' BUFFERSIZE 8
NONBLOCKED MAXFILES 5 MAXFILESIZE 32
```

32 is the maximum number of 4K pages that an event file can contain.

This value must be greater than the value specified by the `BUFFERSIZE` parameter. You can also specify that there is to be no limit on an event file's size:

```
CREATE EVENT MONITOR d1mon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
WRITE TO FILE '/tmp/d1events' BUFFERSIZE 8
NONBLOCKED MAXFILES NONE MAXFILESIZE NONE
```

7. Specify if the event monitor is to be activated automatically each time the database starts. By default, event monitors (with the exception of the WLM event monitors) are not activated automatically when the database starts.
  - To create an event monitor that starts automatically when the database starts, issue the following statement:
 

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
      WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
      NONBLOCKED AUTOSTART
```
  - To create an event monitor that does not start automatically when the database starts, issue the following statement:
 

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
      WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
      NONBLOCKED MANUALSTART
```
8. To activate or deactivate an event monitor, use the SET EVENT MONITOR STATE statement.

Once a file event monitor is created and activated, it will record monitoring data as its specified events occur.

### Event monitor file management

A file event monitor enables the event monitor to store its event records in files. All the output of the event monitor goes in the directory supplied in the FILE parameter for the CREATE EVENT MONITOR statement. This directory will not be created by DB2 if it does not exist. Before the monitor is activated, the directory must exist, or the SET EVENT MONITOR command will return an error. When a file event monitor is first activated, a control file named db2event.ctl is created in this directory. Do not remove or modify this file.

By default, an event monitor writes its trace to a single file, called 00000000.evt. This file keeps growing as long as there is space on the file system. If you specified a file size limit with the MAXFILESIZE parameter of the CREATE EVENT MONITOR statement, then when a file is full, output is automatically directed to the next file. Hence, the active file is the file with the highest number.

You can limit the maximum size of the entire event monitor trace by also using the MAXFILES parameter of the CREATE EVENT MONITOR statement. When the number of files reaches the maximum defined by MAXFILES, the event monitor deactivates itself and the following message is written to the administration notification log.

```
DIA1601I Event Monitor monitor-name was deactivated when it reached
its preset MAXFILES and MAXFILESIZE limit.
```

You can avoid this situation by removing full files. Any event file except the active file can be removed while the event monitor is still running.

If a file event monitor runs out of disk space, it shuts itself down after logging a system-error-level message in the administration notification log.

When a file event monitor is restarted, it can either erase any existing data or append new data to it. This option is specified in the CREATE EVENT MONITOR statement, where either an APPEND monitor or a REPLACE monitor can be created. APPEND is the default option. An APPEND event monitor starts writing at the end of the file it was last using. If you have removed that file, the next file number in sequence is used. When an append event monitor is restarted, only a start\_event is generated. The event log header and database header are generated

only for the first activation. A REPLACE event monitor always deletes existing event files and starts writing at 00000000.evt.

You may want to process monitor data while the event monitor is active. This is possible, and furthermore, when you are finished processing a file, you can delete it, freeing up space for further monitoring data. An event monitor cannot be forced to switch to the next file unless you stop and restart it. It must also be in APPEND mode. In order to keep track of which events have been processed in the active file, you can create an application that simply keeps track of the file number and location of the last record processed. When processing the trace the next time around, the application can simply seek to that file location.

### **Write-to-table and file event monitor buffering**

For write-to-table and file event monitors, the event monitor process buffers its record before writing them to a file or table. Records are written automatically when a buffer is full. Therefore, you can improve monitoring performance for event monitors with high amounts of throughput by specifying larger buffers to reduce the number of disk accesses. To force an event monitor to flush its buffers, you must either deactivate it or empty the buffers by using the FLUSH EVENT MONITOR statement.

A blocked event monitor suspends the database process that is sending monitor data when both of its buffers are full. This is to ensure that no event records are discarded while the blocked event monitor is active. The suspended database process and consequently, any dependent database processes cannot run until a buffer has been written. This can introduce a significant performance overhead, depending on the type of workload and the speed of the I/O device. Event monitors are blocked by default.

A non-blocked event monitor discards monitor data coming from agents when data is coming faster than the event monitor can write the data. This prevents event monitoring from becoming a performance burden on other database activities.

An event monitor that has discarded event records generates an overflow event. It specifies the start and stop time during which the monitor was discarding events and the number of events that were discarded during that period. It is possible for an event monitor to terminate or be deactivated with a pending overflow to report. If this occurs, the following message is written to the admin log:

```
DIA2503I Event Monitor monitor-name had a pending overflow record
when it was deactivated.
```

Loss of event monitoring data can also occur for individual event records. If the length of an event record exceeds the event buffer size, the data that does not fit in the buffer is truncated. For example, this situation could occur if you are capturing the stmt\_text monitor element and applications attached to the database being monitored issue lengthy SQL statements. If you need to capture all of the event record information, specify larger buffers. Keep in mind that larger buffers will result in less frequent writes to file or table.

### **Creating a pipe event monitor**

When creating an event monitor you must determine where the information it collects is to be stored. A pipe event monitor streams event records directly from the event monitor, to a named pipe.

You will need SQLADM or DBADM authority to create a pipe event monitor.

It is the responsibility of the monitoring application to promptly read the data from the pipe as the event monitor writes the event data. If the event monitor is unable to write data to the pipe (for instance, if it is full), monitor data will be lost.

Pipe event monitors are defined with the CREATE EVENT MONITOR statement.

1. Indicate that event monitor data is to be directed to a named pipe.

```
CREATE EVENT MONITOR dlmon FOR eventtype
        WRITE TO PIPE '/home/riihi/dlevents'
```

dlmon is the name of the event monitor.

/home/riihi/dlevents is the name of the named pipe (on UNIX) to where the event monitor will direct the event records. The CREATE EVENT MONITOR statement supports UNIX and Windows pipe naming syntax.

The named pipe specified in the CREATE EVENT MONITOR statement must be present and open when you activate the event monitor. If you specify that the event monitor is to start automatically, the named pipe must exist prior to the event monitor's creation.

2. Specify the types of events to be monitored. You can monitor one or more event types with a single event monitor.

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
        WRITE TO PIPE '/home/riihi/dlevents'
```

This event monitor will monitor for the CONNECTIONS and DEADLOCKS WITH DETAILS event types.

3. Specify if the event monitor is to be activated automatically each time the database starts. By default, event monitors are not activated automatically when the database starts.
  - To create an event monitor that starts automatically when the database starts, issue the following statement:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
        WRITE TO PIPE '/home/riihi/dlevents'
        AUTOSTART
```
  - To create an event monitor that does not start automatically when the database starts, issue the following statement:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
        WRITE TO PIPE '/home/riihi/dlevents'
        MANUALSTART
```
4. To activate or deactivate an event monitor, use the SET EVENT MONITOR STATE statement.

Once a pipe event monitor is created and activated, it will record monitoring data as its specified events occur.

## Event monitor named pipe management

A pipe event monitor enables the processing of the event monitor data stream through a named pipe. Using a pipe event monitor is desirable if you need to process event records in real time. Another important advantage is that your application can ignore unwanted data as it is read off the pipe, giving the opportunity to considerably reduce storage requirements.

On AIX, you can create named pipes by using the mkfifo command. On Linux and other UNIX types (such as the Solaris operating system) use the pipe() routine. On Windows, you can create named pipes by using the CreateNamedPipe() routine.



When you direct data to a pipe, I/O is always blocked and the only buffering is that performed by the pipe. It is the responsibility of the monitoring application to promptly read the data from the pipe as the event monitor writes the event data. If the event monitor is unable to write the data to the pipe (for example, because the pipe is full), monitor data will be lost.

In addition, there must be enough space in the named pipe to handle incoming event records. If the application does not read the data from the named pipe fast enough, the pipe will fill up and overflow. The smaller the pipe buffer, the greater the chance of an overflow.

When a pipe overflow occurs, the monitor creates overflow event records indicating that an overflow has occurred. The event monitor is not turned off, but monitor data is lost. If there are outstanding overflow event records when the monitor is deactivated, a diagnostic message will be logged. Otherwise, the overflow event records will be written to the pipe when possible.

If your operating system allows you to define the size of the pipe buffer, use a pipe buffer of at least 32K. For high-volume event monitors, you should set the monitoring application's process priority equal to or higher than the agent process priority.

## **Creating an event monitor for partitioned databases**

When creating a file or pipe event monitor on partitioned database systems you need to determine the scope of the monitoring data you wish to collect.

### **Before you begin**

You will need SQLADM or DBADM authority to create event monitors for partitioned databases.

### **About this task**

An event monitor uses an operating system process or a thread to write the event records. The database partition where this process or thread runs is called the monitor partition. File and pipe event monitors can be monitoring events as they occur locally on the monitor partition, or globally as they occur on any partition where the DB2 database manager is running. A global event monitor writes a single trace on the monitoring partition that contains activity from all partitions. Whether an event monitor is local or global is referred to as its monitoring scope.

Both the monitor partition and monitor scope are specified with the CREATE EVENT MONITOR statement.

An event monitor can only be activated if the monitor partition is active. If the SET EVENT MONITOR statement is used to activate an event monitor but the monitor partition is not yet active, then event monitor activation will occur when the monitor partition is next started. Furthermore, event monitor activation will automatically occur until the event monitor is explicitly deactivated or the instance is explicitly deactivated. For example, on database partition 0:

```
db2 connect to sample
db2 create event monitor foo ... on dbpartitionnum 2
db2 set event monitor foo state 1
```



After running the above commands, event monitor foo will activate automatically whenever the database sample activates on database partition 2. This automatic activation occurs until `db2 set event monitor foo state 0` is issued, or partition 2 is stopped.

For write-to-table event monitors, the notion of local or global scope is not applicable. When a write-to-table event monitor is activated, an event monitor process runs on all of the partitions. (More specifically, the event monitor process will run on partitions that belong to the database partition groups in which the target tables reside.) Each partition where the event monitor process is running also has the same set of target tables. The data in these tables will be different as it represents the individual partition's view of the monitoring data. You can get aggregate values from all the partitions by issuing SQL statements that access the desired values in each partition's event monitor target tables.

The first column of each target table is named `PARTITION_KEY`, and is used as the partitioning key for the table. The value of this column is chosen so that each event monitor process inserts data into the database partition on which the process is running; that is, insert operations are performed locally on the database partition where the event monitor process is running. On any database partition, the `PARTITION_KEY` field will contain the same value. This means that if a data partition is dropped and data redistribution is performed, all data on the dropped database partition will go to one other database partition instead of being evenly distributed. Therefore, before removing a database partition, consider deleting all table rows on that database partition.

In addition, a column named `PARTITION_NUMBER` can be defined for each table. This column contains the number of the partition on which the data was inserted. Unlike the `PARTITION_KEY` column, the `PARTITION_NUMBER` column is not mandatory.

The table space within which target tables are defined must exist across all partitions that will have event monitor data written to them. Failure to observe this rule will result in records not being written to the log on partitions (with event monitors) where the table space does not exist. Events will still be written on partitions where the table space does exist, and no error will be returned. This behavior allows users to choose a subset of partitions for monitoring, by creating a table space that exists only on certain partitions.

During write-to-table event monitor activation, the `CONTROL` table rows for `FIRST_CONNECT` and `EVMON_START` are only inserted on the catalog database partition. This requires that the table space for the control table exist on the catalog database partition. If it does not exist on the catalog database partition, these inserts are not performed.

If a partition is not yet active when a write-to-table event monitor is activated, the event monitor will be activated when that partition next activates.

If you add a database partition that is online immediately after being added, the event monitors are not immediately aware of the new partition. To collect and record data about the new partition, you must do one of the following:

- For global event monitors, restart the event monitors.
- For write-to-table event monitors, drop, recreate, and restart the event monitors.

**Note:** The lock list in the detailed deadlock connection event will only contain the locks held by the application on the partition where it is waiting for the lock. For example, if an application involved in a deadlock is waiting for a lock on node 20, only the locks held by that application on node 20 will be included in the list.

### Procedure

1. Specify the partition to be monitored.

```
CREATE EVENT MONITOR d1mon FOR DEADLOCKS
    WRITE TO FILE '/tmp/d1events'
    ON PARTITION 3
```

d1mon represents the name of the event monitor.

/tmp/d1events is the name of the directory path (on UNIX) where the event monitor is to write the event files.

3 represents the partition number to be monitored.

2. Specify if the event monitor data is to be collected at a local or global scope. To collect event monitor reports from all partitions issue the following statement:

```
CREATE EVENT MONITOR d1mon FOR DEADLOCKS
    WRITE TO FILE '/tmp/d1events'
    ON PARTITION 3 GLOBAL
```

Only deadlock and deadlock with details event monitors can be defined as GLOBAL. All partitions will report deadlock-related event records to partition 3.

3. To collect event monitor reports from only the local partition issue the following statement:

```
CREATE EVENT MONITOR d1mon FOR DEADLOCKS
    WRITE TO FILE '/tmp/d1events'
    ON PARTITION 3 LOCAL
```

This is the default behavior for file and pipe event monitors in partitioned databases. The LOCAL and GLOBAL clauses are ignored for write-to-table event monitors.

4. It is possible to review the monitor partition and scope values for existing event monitors. To do this query the SYSCAT.EVENTMONITORS table with the following statement:

```
SELECT EVMONNAME, NODENUM, MONSCOPE FROM SYSCAT.EVENTMONITORS
```

### Results

Once an event monitor is created and activated, it will record monitoring data as its specified events occur.

## Event monitor sample output

### Formatting file or pipe event monitor output from a command line

The output of a file or pipe event monitor is a binary stream of logical data groupings. You can format this data stream from a command line by using the db2evmon command. This productivity tool reads in event records from an event monitor's files or pipe, then writes them to the screen (standard output).

No authorization is required unless you are connecting to the database, in which case one of the following is required:

- SYSADM

- SYSCTRL
- SYSMANT
- DBADM

You can indicate which event monitor's output you will format by either providing the path of the event files, or providing the name of the database and the event monitor's name. To format event monitor output:

- Specify the directory containing the event monitor files:

```
db2evmon -path '/tmp/dlevents'
```

`/tmp/dlevents` represents a (UNIX) path.

- Specify the database and event monitor name:

```
db2evmon -db 'sample' -evm 'dlmon'
```

`sample` represents the database the event monitor belongs to.

`dlmon` represents an event monitor.

## Event records and their corresponding applications

In an event trace for an active database with hundreds of attached applications, it can be tedious to track event records associated with a specific application. For traceability, each event record includes the application handle and application ID. These allow you to correlate each record with the application for which the event record was generated.

The application handle (**agent\_id**) is unique system-wide for the duration of the application. However, it will eventually be reused (a 16 bit counter is used to generate this identifier -- on partitioned database systems this consists of the coordinating partition number and a 16 bit counter). In most cases, this reuse is not a problem, since an application reading records from the trace is able to detect a connection that was terminated. For example, encountering (in the trace) a connection header with a known `agent_ID` implies that the previous connection with this `agent_ID` was terminated.

The application ID is a string identifier that includes a timestamp and is guaranteed to remain unique, even after stopping and restarting the database manager.

Finding event records for a certain application is particularly easy with write-to-table event monitors. In the event monitor tables, where each row corresponds to an event record, the application handle and application ID are default column values. To find all the event records for a given application, you can simply issue an SQL select statement for all event records corresponding to the particular application ID.

## Event monitor self-describing data stream

The output of an event monitor is a binary stream of logical data groupings that are exactly the same for both pipe and file event monitors. You can format the data stream either by using the `db2evmon` command or by developing a client application. This data stream is presented in a self-describing format. Figure 1 on page 77 shows the structure of the data stream and Table 37 on page 77 provides some examples of the logical data groups and monitor elements that could be returned.

**Note:** In the examples and tables descriptive names are used for the identifiers. These names are prefixed by `SQLM_ELM_` in the actual data stream. For example,

**db\_event** would appear as **SQLM\_ELM\_DB\_EVENT** in the event monitor output. Types are prefixed with **SQLM\_TYPE\_** in the actual data stream. For example, headers appear as **SQLM\_TYPE\_HEADER** in the data stream.

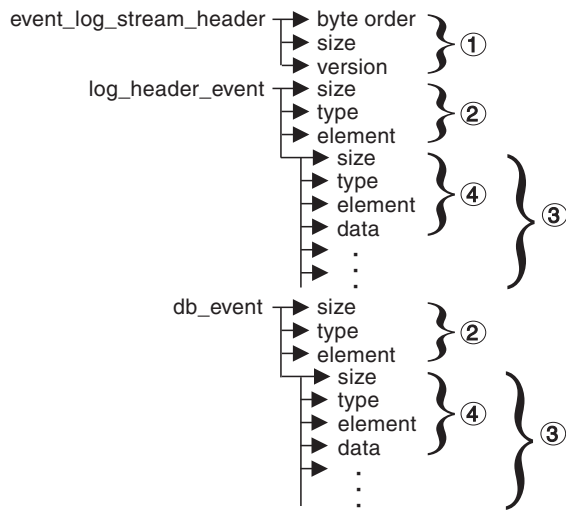


Figure 1. Event Monitor Data Stream

1. The structure of the `sqlm_event_log_data_stream_header` is different than the other headers in the data stream. The version field determines if the output can be processed as a self-describing data stream.

This header has the same size and type as pre-Version 6 event monitor streams. This allows applications to determine if the event monitor output is self-describing or is in the pre-Version 6 static format.

**Note:** This monitor element is extracted by reading `sizeof(sqlm_event_log_data_stream)` bytes from the data stream.

2. Each logical data group begins with a header that indicates its size and element name. This does not apply `event_log_stream_header`, as its size element contains a dummy value to maintain backwards compatibility.
3. The size element in the header indicates the size of all the data in that logical data group.
4. Monitor element information follows its logical data group header and is also self-describing.

Table 37. Sample event data stream

Logical Data Group	Data Stream	Description
event_log_stream_header	sqlm_little_endian	Not used (for compatibility with previous releases).
	200	Not used (for compatibility with previous releases).
	sqlm_dbmon_version9_5	The version of the database manager that returned the data. Event monitors write data in the self-describing format.

Table 37. Sample event data stream (continued)

Logical Data Group	Data Stream	Description
log_header_event	100	Size of the logical data group.
	header	Indicates the start of a logical data group.
	log_header	Name of the logical data group.
	4	Size of the data stored in this monitor element.
	u32bit	Monitor element type - 32 bit numeric.
	byte_order	The name of the monitor element collected.
	little_endian	The collected value for this element.
	2	Size of the data stored in this monitor element.
	u16bit	Monitor element type - unsigned 16 bit numeric.
	codepage_id	The name of the monitor element collected.
	850	The collected value for this element.
db_event	100	Size of the logical data group.
	header	Indicates the start of a logical data group.
	db_event	Name of the logical data group.
	4	Size of the data stored in this monitor element.
	u32bit	Monitor element type - unsigned 32 bit numeric.
	lock_waits	The name of the monitor element collected.
	2	The collected value for this element.

The event\_log\_stream\_header identifies the version of the database manager that returned the data. Event monitors write their data in the self-describing format. An event monitor, unlike a snapshot monitor, does not have a **size** element that returns the total size of the trace. The number present in event\_log\_stream\_header is a dummy value present for backwards compatibility. The total size of an event trace is not known when the event\_log\_stream\_header is written. You typically read an event monitor trace until you reach an end of file or pipe.

The log header describes the characteristics of the trace, containing information such as the memory model (for example little endian) of the server where the trace was collected, and the code page of the database. You might have to do byte swapping on numerical values, if the system where you read the trace has a different memory model than the server (for example, if you are reading a trace from a UNIX server on a Windows 2000 system). Code page translation might also need to be done if the database is configured in a different language than the machine from which you read the trace. When reading the trace, you can use the **size** element to skip a logical data group in the trace.

### Transferring event monitor data between systems

When transferring event monitor information between systems using different conventions for storing numerical values, conversions must be made. Information on UNIX platforms is stored in little endian byte order, and information on Windows platforms is stored in big endian byte order. If event monitor data from a little endian source is to be read on a big endian platform, or vice versa, byte conversion is necessary.

1. To convert the numeric values in logical data group headers and monitor elements use the following logic (presented in C):

```

#include sqlmon.h
#define SWAP2(s) (((s) >> 8) & 0xFF) | (((s) << 8) & 0xFF00)

#define SWAP4(l) (((l) >> 24) & 0xFF) | (((l) & 0xFF0000) >> 8) & 0xFF00 \
                | (((l) & 0xFF00) << 8) | ((l) << 24)

#define SWAP8( where )
{
    sqluint32 temp;
    temp = SWAP4(*(sqluint32 *) (where));
    *(sqluint32 *) (where) = SWAP4(* ((sqluint32 *) (where)) + 1));
    * ((sqluint32 *) (where)) + 1) = temp;
}

int HeaderByteReverse( sqlm_header_info * pHeader)
{
    int rc = 0;

    pHeader->size = SWAP4(pHeader->size);
    pHeader->type = SWAP2(pHeader->type);
    pHeader->element = SWAP2(pHeader->element);

    return rc;
}

int DataByteReverse( char * dataBuf, sqluint32 dataSize)
{
    int rc = 0;

    sqlm_header_info * pElemHeader = NULL;
    char * pElemData = NULL;
    sqluint32 dataOffset = 0;
    sqluint32 elemDataSize = 0;
    sqluint32 elemHeaderSize = sizeof( sqlm_header_info);

    // For each of the elements in the data stream that are numeric,
    // perform byte reversal.

    while( dataOffset < dataSize)
    {
        /* byte reverse the element header */
        pElemHeader = (sqlm_header_info *)
            ( dataBuf + dataOffset);

        rc = HeaderByteReverse( pElemHeader);
        if( rc != 0) return rc;
        // Remember the element data's size...it will be byte reversed
        // before we skip to the next element.
        elemDataSize = pElemHeader->size;

        /* byte reverse the element data */
        pElemData = (char *)
            ( dataBuf + dataOffset + elemHeaderSize);

        if(pElemHeader->type == SQLM_TYPE_HEADER)
        {
            rc = DataByteReverse( pElemData, pElemHeader->size);
            if( rc != 0) return rc;
        }
        else
        {
            switch( pElemHeader->type)
            {
                case SQLM_TYPE_16BIT:
                case SQLM_TYPE_U16BIT:
                    *(sqluint16 *) (pElemData) =
                        SWAP2(*(short *) (pElemData));

                    break;

                case SQLM_TYPE_32BIT:
                case SQLM_TYPE_U32BIT:
                    *(sqluint32 *) (pElemData) =
                        SWAP4(*(sqluint32 *) (pElemData));

                    break;
            }
        }
        dataOffset += elemHeaderSize + elemDataSize;
    }
}

```

```

        case SQLM_TYPE_64BIT:
        case SQLM_TYPE_U64BIT:
            SWAP8(pElemData);
            break;
        default:
            // Not a numeric type. Do nothing.
            break;
    }
}
dataOffset = dataOffset + elemHeaderSize + elemDataSize;
}

return 0;
} /* end of DataByteReverse */

```

2. To convert the numeric values in logical data group headers and monitor elements use the following logic (presented in C):

```

#include sqlmon.h
#define SWAP2(s) (((s) >> 8) & 0xFF) | (((s) << 8) & 0xFF00)

#define SWAP4(l) (((l) >> 24) & 0xFF) | (((l) & 0xFF0000) >> 8) & 0xFF00 \
    | (((l) & 0xFF00) << 8) | ((l) << 24)

#define SWAP8( where )
{
    sqluint32 temp;
    temp = SWAP4(*(sqluint32 *) (where));
    * (sqluint32 *) (where) = SWAP4(* (((sqluint32 *) (where)) + 1));
    * (((sqluint32 *) (where)) + 1) = temp;
}

int HeaderByteReverse( sqlm_header_info * pHeader)
{
    int rc = 0;

    pHeader->size = SWAP4(pHeader->size);
    pHeader->type = SWAP2(pHeader->type);
    pHeader->element = SWAP2(pHeader->element);

    return rc;
}

int DataByteReverse( char * dataBuf, sqluint32 dataSize)
{
    int rc = 0;

    sqlm_header_info * pElemHeader = NULL;
    char * pElemData = NULL;
    sqluint32 dataOffset = 0;
    sqluint32 elemDataSize = 0;
    sqluint32 elemHeaderSize = sizeof( sqlm_header_info);

    // For each of the elements in the data stream that are numeric,
    // perform byte reversal.

    while( dataOffset < dataSize)
    {
        /* byte reverse the element header */
        pElemHeader = (sqlm_header_info *)
            ( dataBuf + dataOffset);

        rc = HeaderByteReverse( pElemHeader);
        if( rc != 0) return rc;
        // Remember the element data's size...it will be byte reversed
        // before we skip to the next element.
        elemDataSize = pElemHeader->size;

        /* byte reverse the element data */
        pElemData = (char *)
            ( dataBuf + dataOffset + elemHeaderSize);
    }
}

```

```

        if(pElemHeader->type == SQLM_TYPE_HEADER)
        {   rc = DataByteReverse( pElemData, pElemHeader->size);
            if( rc != 0) return rc;
        }
        else
        {   switch( pElemHeader->type)
            {   case SQLM_TYPE_16BIT:
                case SQLM_TYPE_U16BIT:
                    *(sqluint16 *) (pElemData) =
                        SWAP2(*(short *) (pElemData));

                    break;
                case SQLM_TYPE_32BIT:
                case SQLM_TYPE_U32BIT:
                    *(sqluint32 *) (pElemData) =
                        SWAP4(*(sqluint32 *) (pElemData));

                    break;
                case SQLM_TYPE_64BIT:
                case SQLM_TYPE_U64BIT:
                    SWAP8(pElemData);
                    break;
                default:
                    // Not a numeric type. Do nothing.
                    break;
            }
        }
        dataOffset = dataOffset + elemHeaderSize + elemDataSize;
    }

    return 0;
} /* end of DataByteReverse */

```





---

## Chapter 4. Snapshot monitor

You can use the snapshot monitor to capture information about the database and any connected applications at a specific time. Snapshots are useful for determining the status of a database system. Taken at regular intervals, they are also useful for observing trends and foreseeing potential problems. Some of the data from the snapshot monitor is obtained from the system monitor. The data available from the system monitor is determined by system monitor switches.

The system monitor accumulates information for a database only while it is active. If all applications disconnect from a database and the database deactivates, then the system monitor data for that database is no longer available. You can keep the database active until your final snapshot has been taken, either by starting the database with the `ACTIVATE DATABASE` command, or by maintaining a permanent connection to the database.

Snapshot monitoring requires an instance attachment. If there is not an attachment to an instance, then a default instance attachment is created. An instance attachment is usually done implicitly to the instance specified by the `DB2INSTANCE` environment variable when the first database system monitor API is invoked by the application. It can also be done explicitly, using the `ATTACH TO` command. Once an application is attached, all system monitor requests that it invokes are directed to that instance. This allows a client to monitor a remote server by simply attaching to the instance on it.

In partitioned database environments, snapshots can be taken at any partition of the instance, or globally using a single instance connection. A global snapshot aggregates the data collected at each partition and returns a single set of values.

You can capture a snapshot from the CLP, from SQL table functions, or by using the snapshot monitor APIs in a C or C++ application. A number of different snapshot request types are available, each returning a specific type of monitoring data. For example, you can capture a snapshot that returns only buffer pool information, or a snapshot that returns database manager information. Before capturing a snapshot, consider if you need information from monitor elements that are under monitor switch control. If a particular monitor switch is off, the monitor elements under its control will not be collected.

---

### Access to system monitor data: SYSMON authority

Users that are part of the SYSMON database manager level group have the authority to gain access to database system monitor data. System monitor data is accessed using the snapshot monitor APIs, CLP commands, or SQL table functions.

The SYSMON authority group replaces the `DB2_SNAPSHOT_NOAUTH` registry variable as the means to enable users without system administration or system control authorities to access database system monitor data.

Aside from SYSMON authority, the only way to access system monitor data using the snapshot monitor is with system administration or system control authority.

Any user that is part of the SYSMON group or has system administration or system control authority can perform the following snapshot monitor functions:

- CLP Commands:
  - GET DATABASE MANAGER MONITOR SWITCHES
  - GET MONITOR SWITCHES
  - GET SNAPSHOT
  - LIST ACTIVE DATABASES
  - LIST APPLICATIONS
  - LIST DCS APPLICATIONS
  - LIST UTILITIES
  - RESET MONITOR
  - UPDATE MONITOR SWITCHES
- APIs:
  - db2GetSnapshot - Get Snapshot
  - db2GetSnapshotSize - Estimate Size Required for db2GetSnapshot() Output Buffer
  - db2MonitorSwitches - Get/Update Monitor Switches
  - db2ResetMonitor - Reset Monitor
- Snapshot SQL table functions without previously running SYSPROC.SNAP\_WRITE\_FILE

---

## Capturing database system snapshots using snapshot administrative views and table functions

Authorized users can capture snapshots of monitor information for a DB2 instance by using snapshot administrative views or snapshot table functions. The snapshot administrative views provide a simple means of accessing data for all database partitions of the connected database. The snapshot table functions allow you to request data for a specific database partition, globally aggregated data, or data from all database partitions. Some snapshot table functions allow you to request data from all active databases.

You must have SYSADM, SYSCTRL, SYSMAINT, or SYSMON authority to capture a database snapshot. To obtain a snapshot of a remote instance, you must first connect to a local database belonging to that instance.

While new snapshot table functions may be required in future releases if new monitor data is available, the set of snapshot administrative views will remain the same with new columns added to the view, making the administrative views a good choice for application maintenance over time.

Each snapshot view returns a table with one row per monitored object per database partition with each column representing a monitor element. Each table function returns a table with one row per monitored object for the specified partition. The column names of the returned table correlate with the monitor element names.

For example, a snapshot of general application information for the SAMPLE database is captured as follows using the SNAPAPPL administrative view:

```
SELECT * FROM SYSIBMADM.SNAPAPPL
```

You can also select individual monitor elements from the returned table. For example, the following statement returns only the **agent\_id** and **appl\_id** monitor elements:

```
SELECT agent_id, appl_id FROM SYSIBMADM.SNAPAPPL
```

Snapshot administrative views and table functions cannot be used in conjunction with either of the following:

- Monitor switches commands/APIs
- Monitor reset commands/APIs

This restriction includes:

- GET MONITOR SWITCHES
- UPDATE MONITOR SWITCHES
- RESET MONITOR

This limitation is due to the fact that such commands use an INSTANCE ATTACH, while snapshot table functions make use of DATABASE CONNECTs.

To capture a snapshot using a snapshot administrative view:

1. To capture a snapshot using a snapshot administrative view:
  - a. Connect to a database. This can be any database in the instance you need to monitor. To be able to issue an SQL query with a snapshot administrative view, you must be connected to a database.
  - b. Determine the type of snapshot you need to capture. If you want to capture a snapshot for a database other than the currently connected database, or if you want to retrieve data from a single database partition, or global aggregate data, you need to use a snapshot table function instead.
  - c. Issue a query with the appropriate snapshot administrative view. For example, here is a query that captures a snapshot of lock information for the currently connected database:

```
SELECT * FROM SYSIBMADM.SNAPLOCK
```

2. To capture a snapshot using a snapshot table function:
  - a. Connect to a database. This can be any database in the instance you need to monitor. To be able to issue an SQL query with a snapshot table function, you must be connected to a database.
  - b. Determine the type of snapshot you need to capture.
  - c. Issue a query with the appropriate snapshot table function. For example, here is a query that captures a snapshot of lock information about the SAMPLE database for the current connected database partition:

```
SELECT * FROM TABLE(SNAP_GET_LOCK('SAMPLE',-1)) AS SNAPLOCK
```

The SQL table functions have two input parameters:

**database name**

VARCHAR(255). If you enter NULL, the name of the currently connected database is used.

**partition number**

SMALLINT. For the database partition number parameter, enter the integer (a value between 0 and 999) corresponding to the database partition number you need to monitor. To capture a snapshot for the currently connected database partition, enter a value of -1. To capture a global aggregate snapshot, enter a value of -2. To capture a snapshot from all database partitions, do not specify a value for this parameter.

**Note:**

- 1) For the following list of snapshot table functions, if you enter a NULL for the currently connected database, you will get snapshot information for all databases in the instance:
  - SNAP\_GET\_DB\_V95
  - SNAP\_GET\_DB\_MEMORY\_POOL
  - SNAP\_GET\_DETAILLOG\_V91
  - SNAP\_GET\_HADR
  - SNAP\_GET\_STORAGE\_PATHS
  - SNAP\_GET\_APPL\_V95
  - SNAP\_GET\_APPL\_INFO\_V95
  - SNAP\_GET\_AGENT
  - SNAP\_GET\_AGENT\_MEMORY\_POOL
  - SNAP\_GET\_STMT
  - SNAP\_GET\_SUBSECTION
  - SNAP\_GET\_BP\_V95
  - SNAP\_GET\_BP\_PART
- 2) The database name parameter does not apply to the database manager level snapshot table functions; they have only a parameter for database partition number. The database partition number parameter is optional.

---

## Capturing database system snapshot information to a file using the SNAP\_WRITE\_FILE stored procedure

With the SNAP\_WRITE\_FILE stored procedure you can capture snapshots of monitor data and save this information to files on the database server and allow access to the data by users who do not have SYSADM, SYSCtrl, SYSMAINT, or SYSMON authority. Any user can then issue a query with a snapshot table function to access the snapshot information in these files. In providing open access to snapshot monitor data, sensitive information (such as the list of connected users and the SQL statements they have submitted to the database) is available to all users who have the execution privilege for the snapshot table functions. The privilege to execute the snapshot table functions is granted to PUBLIC by default. (Note, however, that no actual data from tables or user passwords can be exposed using the snapshot monitor table functions.)

You must have SYSADM, SYSCtrl, SYSMAINT, or SYSMON authority to capture a database snapshot with the SNAP\_WRITE\_FILE stored procedure.

When issuing a call to the SNAP\_WRITE\_FILE stored procedure, in addition to identifying the database and partition to be monitored, you need to specify a *snapshot request type*. Each snapshot request type determines the scope of monitor data that is collected. Choose the snapshot request types based on the snapshot table functions users will need to run. The following table lists the snapshot table functions and their corresponding request types.

*Table 38. Snapshot request types*

Snapshot table function	Snapshot request type
SNAP_GET_AGENT	APPL_ALL
SNAP_GET_AGENT_MEMORY_POOL	APPL_ALL

Table 38. Snapshot request types (continued)

Snapshot table function	Snapshot request type
SNAP_GET_APPL_V95	APPL_ALL
SNAP_GET_APPL_INFO_V95	APPL_ALL
SNAP_GET_STMT	APPL_ALL
SNAP_GET_SUBSECTION	APPL_ALL
SNAP_GET_BP_PART	BUFFERPOOLS_ALL
SNAP_GET_BP_V95	BUFFERPOOLS_ALL
SNAP_GET_DB_V95	DBASE_ALL
SNAP_GET_DETAILLOG_V91	DBASE_ALL
SNAP_GET_DB_MEMORY_POOL	DBASE_ALL
SNAP_GET_HADR	DBASE_ALL
SNAP_GET_STORAGE_PATHS	DBASE_ALL
SNAP_GET_DBM_V95	DB2
SNAP_GET_DBM_MEMORY_POOL	DB2
SNAP_GET_FCM	DB2
SNAP_GET_FCM_PART	DB2
SNAP_GET_SWITCHES	DB2
SNAP_GET_DYN_SQL_V95	DYNAMIC_SQL
SNAP_GET_LOCK	DBASE_LOCKS
SNAP_GET_LOCKWAIT	APPL_ALL
SNAP_GET_TAB_V91	DBASE_TABLES
SNAP_GET_TAB_REORG	DBASE_TABLES
SNAP_GET_TBSP_V91	DBASE_TABLESPACES
SNAP_GET_TBSP_PART_V91	DBASE_TABLESPACES
SNAP_GET_CONTAINER_V91	DBASE_TABLESPACES
SNAP_GET_TBSP QUIESCER	DBASE_TABLESPACES
SNAP_GET_TBSP_RANGE	DBASE_TABLESPACES
SNAP_GET_UTIL	DB2
SNAP_GET_UTIL_PROGRESS	DB2

1. Connect to a database. This can be any database in the instance you need to monitor. To be able to call a stored procedure, you must be connected to a database.
2. Determine the snapshot request type, and the database and partition you need to monitor.
3. Call the SNAP\_WRITE\_FILE stored procedure with the appropriate parameter settings for the snapshot request type, database, and partition. For example, here is a call that will capture a snapshot of application information about the SAMPLE database for the current connected partition:

```
CALL SNAP_WRITE_FILE('APPL_ALL', 'SAMPLE', -1)
```

The SNAP\_WRITE\_FILE stored procedure has three input parameters:

- a snapshot request type (see Table 38 on page 86, which provides a cross-reference of the snapshot table functions and their corresponding request types)
- a VARCHAR (128) for the database name. If you enter NULL, the name of the currently connected database is used.

**Note:** This parameter does not apply to the database manager level snapshot table functions; they only have parameters for request type and partition number.

- a SMALLINT for the partition number (a value between 0 and 999). For the partition number parameter, enter the integer corresponding to partition number you wish to monitor. To capture a snapshot for the currently connected partition, enter a value of -1 or a NULL. To capture a global snapshot, enter a value of -2.

Once the snapshot data has been saved to a file, all users can issue queries with the corresponding snapshot table functions, specifying (NULL, NULL) as input values for database-level table functions, and (NULL) for database manager level table functions. The monitor data they receive is pulled from the files generated by the SNAP\_WRITE\_FILE stored procedure.

**Note:** While this provides a means to limit user access to sensitive monitor data, this approach does have some limitations:

- The snapshot monitor data available from the SNAP\_WRITE\_FILE files is only as recent as the last time the SNAP\_WRITE\_FILE stored procedure was called. You can ensure that recent snapshot monitor data is available by making calls to the SNAP\_WRITE\_FILE stored procedure at regular intervals. For instance, on UNIX systems you can set a cron job to do this.
- Users issuing queries with the snapshot table functions cannot identify a database or partition to monitor. The database name and partition number identified by the user issuing the SNAP\_WRITE\_FILE calls determine the contents of the files accessible by the snapshot table functions.
- If a user issues an SQL query containing a snapshot table function for which a corresponding SNAP\_WRITE\_FILE request type has not been run, a direct snapshot is attempted for the currently connected database and partition. This operation is successful only if the user has SYSADM, SYSCTRL, SYSMOINT, or SYSMON authority.

---

## Accessing database system snapshots using snapshot table functions in SQL queries (with file access)

For every request type that authorized users have called the SNAP\_WRITE\_FILE stored procedure, any user can issue queries with the corresponding snapshot table functions. The monitor data they receive will be retrieved from the files generated by the SNAP\_WRITE\_FILE stored procedure.

For every snapshot table function with which you intend to access SNAP\_WRITE\_FILE files, an authorized user must have issued a SNAP\_WRITE\_FILE stored procedure call with the corresponding snapshot request types. If you issue an SQL query containing a snapshot table function for which a corresponding SNAP\_WRITE\_FILE request type has not been run, a direct snapshot is attempted for the currently connected database and partition. This operation is successful only if the user has SYSADM, SYSCTRL, SYSMOINT, or SYSMON authority.

Users who access snapshot data from SNAP\_WRITE\_FILE files with snapshot table functions cannot identify a database or partition to monitor. The database name and partition number identified by the user issuing the SNAP\_WRITE\_FILE calls determine the contents of the SNAP\_WRITE\_FILE files. The snapshot monitor data available from the SNAP\_WRITE\_FILE files is only as recent as the last time the SNAP\_WRITE\_FILE stored procedure captured snapshots.

1. Connect to a database. This can be any database in the instance you need to monitor. To issue an SQL query with a snapshot table function, you must be connected to a database.
2. Determine the type of snapshot you need to capture.
3. Issue a query with the appropriate snapshot table function. For example, here is a query that will capture a snapshot of table space information:

```
SELECT * FROM TABLE(SNAP_GET_TBSP_V91 (CAST(NULL AS VARCHAR(1)),
                                       CAST(NULL AS INTEGER))) AS SNAP_GET_TBSP_V91
```

**Note:** You must enter NULL values for the database name and partition number parameters. The database name and partition for the snapshot are determined in the call of the SNAP\_WRITE\_FILE stored procedure. Also, the database name parameter does not apply to the database manager level snapshot table functions; they only have a parameter for partition number. Each snapshot table function returns a table with one or more rows, with each column representing a monitor element. Accordingly, the monitor element column names correlate to the monitor element names.

4. You can also select individual monitor elements from the returned table. For example, the following statement will return only the **agent\_id** monitor element:

```
SELECT agent_id FROM TABLE(
    SNAP_GET_APPL_V95(CAST(NULL AS VARCHAR(1)),
                     CAST(NULL AS INTEGER)))
as SNAP_GET_APPL_V95
```

---

## Snapshot monitor SQL Administrative Views

There are a number of different snapshot monitor SQL administrative views available, each returning monitor data about a specific area of the database system. For example, the SYSIBMADM.SNAPBP SQL administrative view captures a snapshot of buffer pool information. The following table lists each available snapshot monitor administrative view.

*Table 39. Snapshot Monitor SQL Administrative Views*

Monitor level	SQL Administrative Views	Information returned
Database manager	SYSIBMADM.SNAPDBM	Database manager level information.
Database manager	SYSIBMADM.SNAPFCM	Database manager level information regarding the fast communication manager (FCM).
Database manager	SYSIBMADM.SNAPFCM_PART	Database manager level information for a partition regarding the fast communication manager (FCM).
Database manager	SYSIBMADM.SNAPSWITCHES	Database manager monitor switch settings.
Database manager	SYSIBMADM.SNAPDBM_MEMORY_POOL	Database manager level information about memory usage.



Table 39. Snapshot Monitor SQL Administrative Views (continued)

Monitor level	SQL Administrative Views	Information returned
Database	SYSIBMADM.SNAPDB	Database level information and counters for a database. Information is returned only if there is at least one application connected to the database.
Database	SYSIBMADM.SNAPDB_MEMORY_POOL	Database level information about memory usage for UNIX platforms only.
Database	SYSIBMADM.SNAPHADR	Database level information about high availability disaster recovery.
Application	SYSIBMADM.SNAPAPPL	General application level information for each application that is connected to the database. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	SYSIBMADM.SNAPAPPL_INFO	General application level identification information for each application that is connected to the database.
Application	SYSIBMADM.SNAPLOCKWAIT	Application level information regarding lock waits for the applications connected to the database.
Application	SYSIBMADM.SNAPSTMT	Application level information regarding statements for the applications connected to the database. This includes the most recent SQL statement executed (if the statement switch is set).
Application	SYSIBMADM.SNAPAGENT	Application level information regarding the agents associated with applications connected to the database.
Application	SYSIBMADM.SNAPSUBSECTION	Application level information regarding the subsections of access plans for the applications connected to the database.
Application	SYSIBMADM.SNAPAGENT_MEMORY_POOL	Information about memory usage at the agent level.
Table	SYSIBMADM.SNAPTAB	Table activity information at the database and application level for each application connected to the database. Table activity information at the table level for each table that <i>was accessed</i> by an application connected to the database. Requires the table switch.
Table	SYSIBMADM.SNAPTAB_REORG	Table reorganization information at the table level for each table in the database undergoing reorganization.
Lock	SYSIBMADM.SNAPLOCK	Lock information at the database level, and application level for each application connected to the database. Requires the lock switch.
Table space	SYSIBMADM.SNAPTbsp	Information about table space activity at the database level, the application level for each application connected to the database, and the table space level for each table space that has been accessed by an application connected to the database. Requires the buffer pool switch.

Table 39. Snapshot Monitor SQL Administrative Views (continued)

Monitor level	SQL Administrative Views	Information returned
Table space	SYSIBMADM.SNAPTbsp_PART	Information about table space configuration.
Table space	SYSIBMADM.SNAPTbsp_QUIESCER	Information about quiescers at the table space level.
Table space	SYSIBMADM.SNAPCONTAINER	Information about table space container configuration at the table space level.
Table space	SYSIBMADM.SNAPTbsp_RANGE	Information about ranges for a table space map.
Buffer pool	SYSIBMADM.SNAPBP	Buffer pool activity counters for the specified database. Requires the buffer pool switch.
Buffer pool	SYSIBMADM.SNAPBP_PART	Information on buffer size and usage, calculated per partition.
Dynamic SQL	SYSIBMADM.SNAPDYN_SQL	Point-in-time statement information from the SQL statement cache for the database.
Database	SYSIBMADM.SNAPUTIL	Information about utilities.
Database	SYSIBMADM.SNAPUTIL_PROGRESS	Information about the progress of utilities.
Database	SYSIBMADM.SNAPDETAILLOG	Database level information about log files.
Database	SYSIBMADM.SNAPSTORAGE_PATHS	Returns a list of automatic storage paths for the database including file system information for each storage path.

Before capturing a snapshot, consider if you need information from monitor elements that are under monitor switch control. If a particular monitor switch is off, the monitor elements under its control will not be collected. See the individual monitor elements to determine if an element you need is under switch control.

All snapshot monitoring administrative views and associated table functions use a separate instance connection, which is different from the connection the current session uses. Therefore, only default database manager monitor switches are effective. Ineffective monitor switches include any that are turned on or off dynamically from the current session or application.

DB2 Version 9.5 also provides you with a set of administrative views that do not only return values of individual monitor elements, but also return computed values that are commonly required in monitoring tasks. For example, the SYSIBMADM.BP\_HITRATIO administrative view returns calculated values for buffer pool hit ratios, which combine a number of individual monitor elements.

Table 40. Snapshot Monitor SQL Administrative Convenience Views

SQL Administrative Convenience Views	Information returned
SYSIBMADM.APPLICATIONS	Information about connected database applications.
SYSIBMADM.APPL_PERFORMANCE	Information about the rate of rows selected versus the number of rows read by an application.
SYSIBMADM.BP_HITRATIO	Buffer pool hit ratios, including total, data, and index, in the database.
SYSIBMADM.BP_READ_IO	Information about buffer pool read performance.
SYSIBMADM.BP_WRITE_IO	Information about buffer pool write performance.
SYSIBMADM.CONTAINER_UTILIZATION	Information about table space containers and utilization rates.
SYSIBMADM.LOCKS_HELD	Information on current locks held.

Table 40. Snapshot Monitor SQL Administrative Convenience Views (continued)

SQL Administrative Convenience Views	Information returned
ISYSIBMADM.LOCKWAIT	Information about DB2 agents working on behalf of applications that are waiting to obtain locks.
SYSIBMADM.LOG_UTILIZATION	Information about log utilization for the currently connected database.
SYSIBMADM.LONG_RUNNING_SQL	Information about the longest running SQL in the currently connected database.
SYSIBMADM.QUERY_PREP_COST	Information about the time required to prepare different SQL statements.
SYSIBMADM.TBSP_UTILIZATION	Table space configuration and utilization information.
SYSIBMADM.TOP_DYNAMIC_SQL	The top dynamic SQL statements sortable by number of executions, average execution time, number of sorts, or sorts per statement.

## SQL access to database system snapshots

There are two ways to access snapshot monitor data with the snapshot monitor SQL table functions (referred to as *snapshot table functions*):

- direct access
- file access

### Direct access

Authorized users can issue queries with snapshot table functions and receive result sets containing monitor data. With this approach, access to snapshot monitor data is only available to users that have SYSADM, SYSCTRL, SYSMANT, or SYSMON authority.

To capture snapshot information using direct access:

1. Optional: Set and check the status of the monitor switches.
2. Capture database system snapshots using SQL.

### File access

Authorized users call the SNAPSHOT\_FILEW stored procedure, identifying the snapshot request type, and the affected partition and database. The SNAPSHOT\_FILEW stored procedure then saves the monitor data into a file on the database server.

Every request type for which authorized users can call the SNAPSHOT\_FILEW stored procedure,

While this is a safe means of providing all users with access to snapshot monitor data, there are limitations to this approach:

- The snapshot monitor data available from the SNAPSHOT\_FILEW files is only as recent as the last time the SNAPSHOT\_FILEW stored procedure was called. You can ensure that recent snapshot monitor data is available by making calls to the SNAPSHOT\_FILEW stored procedure at regular intervals. For instance, on UNIX systems you can set a cron job to do this.
- Users issuing queries with the snapshot table functions cannot identify a database or partition to monitor. The database name and partition number identified by the user issuing the SNAPSHOT\_FILEW calls determine the contents of the files accessible by the snapshot table functions.

- If a user issues an SQL query containing a snapshot table function for which a corresponding SNAPSHOT\_FILEW request type has not been run, a direct snapshot is attempted for the currently connected database and partition. This operation is successful only if the user has SYSADM, SYSCtrl, SYSMAINT, or SYSMON authority.

The following tasks are performed by the SYSADM, SYSCtrl, SYSMAINT, or SYSMON user who captures database system snapshot information to a file.

1. Find out the needs of users who will issue snapshot requests. Specifically, determine the monitor data they need, the database it is to be collected from, and if the collection needs to be limited to a particular partition.
2. Optional: Set and check the status of the monitor switches.
3. Capture database system snapshot information to a file .

Once the SYSADM, SYSCtrl, SYSMAINT, or SYSMON user has completed the preceding steps, all users can access database system snapshot information using snapshot table functions in SQL queries.

---

## Capturing a database snapshot from the CLP

You can capture database snapshots from the CLP using the GET SNAPSHOT command. A number of different snapshot request types are available, which can be accessed by specifying certain parameters for the GET SNAPSHOT command.

You must have SYSADM, SYSCtrl, SYSMAINT, or SYSMON authority to capture a database snapshot.

You must have an instance attachment to capture a database snapshot. If there is not an attachment to an instance, then a default instance attachment is created. To obtain a snapshot of a remote instance, you must first attach to that instance.

1. Optional: Set and check the status of the monitor switches.
2. From the CLP, issue the GET SNAPSHOT command with the desired parameters. In the following example, a snapshot captures database manager level information:

```
db2 get snapshot for dbm
```

3. For partitioned database systems, you can capture a database snapshot specifically for a certain partition, or globally for all partitions. To capture a database snapshot for all applications on a specific partition (for example, partition number 2), issue the following command:

```
db2 get snapshot for all applications at dbpartitionnum 2
```

4. To capture a database snapshot for all applications on all partitions, issue the following command:

```
db2 get snapshot for all applications global
```

For global snapshots on partitioned databases, the monitor data from all the partitions is aggregated.

---

## Snapshot monitor CLP commands

The following table lists all the supported snapshot request types. For certain request types, some information is returned only if the associated monitor switch is set ON. See the individual monitor elements to determine if a required element is under switch control.

Table 41. Snapshot Monitor CLP Commands

Monitor level	CLP command	Information returned
Connections list	<code>list applications [show detail]</code>	Application identification information for all applications currently connected to a database that is managed by the DB2 instance on the partition where snapshot is taken.
Connections list	<code>list applications for database <i>dbname</i> [show detail]</code>	Application identification information for each application currently connected to the specified database.
Connections list	<code>list dcs applications</code>	Application identification information for all DCS applications currently connected to a database that is managed by the DB2 instance on the partition where snapshot is taken.
Database manager	<code>get snapshot for dbm</code>	Database manager level information, including instance-level monitor switch settings.
Database manager	<code>get dbm monitor switches</code>	Instance-level monitor switch settings.
Database	<code>get snapshot for database on <i>dbname</i></code>	Database level information and counters for a database. Information is returned only if there is at least one application connected to the database.
Database	<code>get snapshot for all databases</code>	Database level information and counters for each database active on the partition. Information is returned only if there is at least one application connected to the database.
Database	<code>list active databases</code>	The number of connections to each active database. Includes databases that were started using the <code>ACTIVATE DATABASE</code> command, but have no connections.
Database	<code>get snapshot for dcs database on <i>dbname</i></code>	Database level information and counters for a specific DCS database. Information is returned only if there is at least one application connected to the database.
Database	<code>get snapshot for remote database on <i>dbname</i></code>	Database level information and counters for a specific federated system database. Information is returned only if there is at least one application connected to the database.
Database	<code>get snapshot for all remote databases</code>	Database level information and counters for each active federated system database on the partition. Information is returned only if there is at least one application connected to the database.
Application	<code>get snapshot for application applid <i>appl-id</i></code>	Application level information, including cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	<code>get snapshot for application agentid <i>appl-handle</i></code>	Application level information, includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	<code>get snapshot for applications on <i>dbname</i></code>	Application level information for each application that is connected to the database on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	<code>get snapshot for all applications</code>	Application level information for each application that is active on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).

Table 41. Snapshot Monitor CLP Commands (continued)

Monitor level	CLP command	Information returned
Application	get snapshot for dcs application applid <i>appl-id</i>	Application level information, including cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	get snapshot for all dcs applications	Application level information for each DCS application that is active on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	get snapshot for dcs application agentid <i>appl-handle</i>	Application level information, including cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	get snapshot for dcs applications on <i>dbname</i>	Application level information for each DCS application that is connected to the database on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	get snapshot for remote applications on <i>dbname</i>	Application level information, includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Application	get snapshot for all remote applications	Application level information for each federated system application that is active on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).
Table	get snapshot for tables on <i>dbname</i>	Table activity information at the database and application level for each application connected to the database. Table activity information at the table level for each table that <b>was accessed</b> by an application connected to the database. Requires the table switch.
Lock	get snapshot for locks for application applid <i>appl-id</i>	List of locks held by the application. Lock wait information requires the lock switch.
Lock	get snapshot for locks for application agentid <i>appl-handle</i>	List of locks held by the application. Lock wait information requires the lock switch.
Lock	get snapshot for locks on <i>dbname</i>	Lock information at the database level, and application level for each application connected to the database. Requires the lock switch.
Table space	get snapshot for tablespaces on <i>dbname</i>	Information about table space activity for a database. Requires the buffer pool switch. Also included is information on containers, quiescers, and ranges. This information is not under switch control.
Buffer pool	get snapshot for all bufferpools	Buffer pool activity counters. Requires the buffer pool switch.
Buffer pool	get snapshot for bufferpools on <i>dbname</i>	Buffer pool activity counters for the specified database. Requires the buffer pool switch.
Dynamic SQL	get snapshot for dynamic sql on <i>dbname</i>	Point-in-time statement information from the SQL statement cache for the database. The information can also be from a remote data source.



---

## Capturing a database snapshot from a client application

You can capture database snapshots using the snapshot monitor API in a C, C++, or a COBOL application. In C and C++ a number of different snapshot request types can be accessed by specifying certain parameters in `db2GetSnapshot()`.

You must have `SYSADM`, `SYSCTRL`, `SYSMAINT`, or `SYSMON` authority to use the `db2MonitorSwitches` API.

You must have an instance attachment to capture a database snapshot. If there is not an attachment to an instance, then a default instance attachment is created. To obtain a snapshot of a remote instance, you must first attach to that instance.

1. Optional: Set and check the status of the monitor switches.
2. Include the following DB2 libraries: `sqlmon.h` and `db2ApiDf.h`. These are found in the include subdirectory under `sqllib`.

```
#include <db2ApiDf.h>
#include <sqlmon.h>
```

3. Set snapshot buffer unit size to 100 KB.
4. Declare the `sqlca`, `sqlma`, `db2GetSnapshotData`, and `sqlm_collected` structures. Also, initialize a pointer to contain the snapshot buffer, and establish the buffer's size.

```
struct sqlma *pRequestedDataGroups;
struct sqlca sqlca;
memset (&sqlca, '\0', sizeof(struct sqlca));
struct sqlm_collected collectedData;
memset (&collectedData, '\0', sizeof(collectedData));
db2GetSnapshotData getSnapshotParam;
memset (&getSnapshotParam, '\0', sizeof(getSnapshotParam));

static sqluint32 snapshotBufferSize = SNAPSHOT_BUFFER_UNIT_SZ;
sqluint32 outputFormat;
char *snapshotBuffer;
```

5. Initialize the `sqlma` structure and specify that the snapshot to be captured is of database manager level information.

```
pRequestedDataGroups = (struct sqlma *)malloc(SQLMASIZE(1));
memset(pRequestedDataGroups, '\0', SQLMASIZE(1));
pRequestedDataGroups->obj_num = 1;
pRequestedDataGroups->obj_var[0].obj_type = SQLMA_DB2;
```

6. Initialize the buffer which is to hold the snapshot output.

```
snapshotBuffer = (char *)malloc(snapshotBufferSize);
memset (snapshotBuffer, '\0', snapshotBufferSize);
```

7. Populate the `db2GetSnapshotData` structure with the snapshot request type (from the `sqlma` structure), buffer information, and other information required to capture a snapshot.

```
getSnapshotParam.piSqlmaData = pRequestedDataGroups;
getSnapshotParam.poCollectedData = &collectedData;
getSnapshotParam.poBuffer = snapshotBuffer;
getSnapshotParam.iVersion = SQLM_DBMON_VERSION9_5;
getSnapshotParam.iBufferSize = snapshotBufferSize;
getSnapshotParam.iStoreResult = 0;
getSnapshotParam.iNodeNumber = SQLM_CURRENT_NODE;
getSnapshotParam.poOutputFormat = &outputFormat;
getSnapshotParam.iSnapshotClass = SQLM_CLASS_DEFAULT;
```

8. Capture the snapshot. Pass the `db2GetSnapshotData` structure, which contains the information necessary to capture a snapshot, as well as a reference to the buffer, where snapshot output is to be directed.

- ```

db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);

```
9. Include logic to handle buffer overflow. After a snapshot is taken, the sqlcode is checked for a buffer overflow. If a buffer overflow occurred the buffer is cleared and reinitialized, and the snapshot is taken again.

```

while (sqlca.sqlcode == 1606)
{
    free(snapshotBuffer);
    snapshotBufferSize = snapshotBufferSize +
    SNAPSHOT_BUFFER_UNIT_SZ;
    snapshotBuffer = (char *)malloc(snapshotBufferSize);
    if (snapshotBuffer == NULL)
    {
        printf("\nMemory allocation error.\n");
        return 1;
    }
    getSnapshotParam.iBufferSize = snapshotBufferSize;
    getSnapshotParam.poBuffer = snapshotBuffer;
    db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
}

```
  10. Process the snapshot monitor data stream.
  11. Clear the buffer.

```

free(snapshotBuffer);
free(pRequestedDataGroups);

```

---

## Snapshot monitor API request types

The following table lists all the supported snapshot request types. For certain request types, some information is returned only if the associated monitor switch is set ON. See the individual monitor elements to determine if a required element is under switch control.

*Table 42. Snapshot Monitor API Request Types*

| Monitor level    | API request type       | Information returned                                                                                                                                                                                                                                                                                                                  |
|------------------|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Connections list | SQLMA_APPLINFO_ALL     | Application identification information for all applications currently connected to a database that is managed by the DB2 instance on the partition where snapshot is taken.                                                                                                                                                           |
| Connections list | SQLMA_DBASE_APPLINFO   | Application identification information for each application currently connected to the specified database.                                                                                                                                                                                                                            |
| Connections list | SQLMA_DCS_APPLINFO_ALL | Application identification information for all DCS applications currently connected to a database that is managed by the DB2 instance on the partition where snapshot is taken.                                                                                                                                                       |
| Database manager | SQLMA_DB2              | Database manager level information, including instance-level monitor switch settings.                                                                                                                                                                                                                                                 |
| Database         | SQLMA_DBASE            | Database level information and counters for a database. Information is returned only if there is at least one application connected to the database.                                                                                                                                                                                  |
| Database         | SQLMA_DBASE_ALL        | Database level information and counters for each database active on the partition. The number of connections to each active database. Includes databases that were started using the ACTIVATE DATABASE command, but have no connections. Information is returned only if there is at least one application connected to the database. |



Table 42. Snapshot Monitor API Request Types (continued)

| Monitor level | API request type       | Information returned                                                                                                                                                                                                                   |
|---------------|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Database      | SQLMA_DCS_DBASE        | Database level information and counters for a specific DCS database. Information is returned only if there is at least one application connected to the database.                                                                      |
| Database      | SQLMA_DCS_DBASE_ALL    | Database level information and counters for each DCS database active on the partition. Information is returned only if there is at least one application connected to the database.                                                    |
| Database      | SQLMA_DBASE_REMOTE     | Database level information and counters for a specific federated system database. Information is returned only if there is at least one application connected to the database.                                                         |
| Database      | SQLMA_DBASE_REMOTE_ALL | Database level information and counters for each active federated system database on the partition. Information is returned only if there is at least one application connected to the database.                                       |
| Application   | SQLMA_APPL             | Application level information, including cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).                                                                                 |
| Application   | SQLMA_AGENT_ID         | Application level information, includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).                                                                                  |
| Application   | SQLMA_DBASE_APPLS      | Application level information for each application that is connected to the database on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).     |
| Application   | SQLMA_APPL_ALL         | Application level information for each application that is active on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).                        |
| Application   | SQLMA_DCS_APPL         | Application level information, including cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).                                                                                 |
| Application   | SQLMA_DCS_APPL_ALL     | Application level information for each DCS application that is active on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).                    |
| Application   | SQLMA_DCS_APPL_HANDLE  | Application level information, including cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).                                                                                 |
| Application   | SQLMA_DCS_DBASE_APPLS  | Application level information for each DCS application that is connected to the database on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set). |

Table 42. Snapshot Monitor API Request Types (continued)

| Monitor level | API request type          | Information returned                                                                                                                                                                                                                                                                    |
|---------------|---------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Application   | SQLMA_DBASE_APPLS_REMOTE  | Application level information, includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).                                                                                                                                   |
| Application   | SQLMA_APPL_REMOTE_ALL     | Application level information for each federated system application that is active on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set).                                                        |
| Table         | SQLMA_DBASE_TABLES        | Table activity information at the database and application level for each application connected to the database. Table activity information at the table level for each table that <b>was accessed</b> by an application connected to the database. Requires the table switch.          |
| Lock          | SQLMA_APPL_LOCKS          | List of locks held by the application. Lock wait information requires the lock switch.                                                                                                                                                                                                  |
| Lock          | SQLMA_APPL_LOCKS_AGENT_ID | List of locks held by the application. Lock wait information requires the lock switch.                                                                                                                                                                                                  |
| Lock          | SQLMA_DBASE_LOCKS         | Lock information at the database level, and application level for each application connected to the database. Requires the lock switch.                                                                                                                                                 |
| Table space   | SQLMA_DBASE_TABLESPACES   | Information about table space activity at the database level, the application level for each application connected to the database, and the table space level for each table space that has been accessed by an application connected to the database. Requires the buffer pool switch. |
| Buffer pool   | SQLMA_BUFFERPOOLS_ALL     | Buffer pool activity counters. Requires the buffer pool switch.                                                                                                                                                                                                                         |
| Buffer pool   | SQLMA_DBASE_BUFFERPOOLS   | Buffer pool activity counters for the specified database. Requires the buffer pool switch.                                                                                                                                                                                              |
| Dynamic SQL   | SQLMA_DYNAMIC_SQL         | Point-in-time statement information from the SQL statement cache for the database.                                                                                                                                                                                                      |

## Snapshot monitor sample output

To illustrate the nature of the snapshot monitor, here is an example of a snapshot being taken using the CLP, along with its corresponding output. The objective in this example is to obtain a list of the locks held by applications connected to the SAMPLE database. The steps taken are as follows:

1. Connect to the sample database:  
db2 connect to sample
2. Turn on the LOCK switch with the UPDATE MONITOR SWITCHES command, so that the time spent waiting for locks is collected:  
db2 update monitor switches using LOCK on
3. Issue a command or statement that will require locks on the database catalogs. In this case, we will declare, open, and fetch a cursor:

```

db2 -c- declare c1 cursor for
                        select * from staff where job='Sales' for update
db2 -c- open c1
db2 -c- fetch c1

```

4. Take the database lock snapshot, using the GET SNAPSHOT command:

```
db2 get snapshot for locks on sample
```

After the GET SNAPSHOT command is issued from the CLP, the snapshot output is directed to the screen.

Database Lock Snapshot

```

Database name           = SAMPLE
Database path          = C:\DB2\NODE0000\SQL00001\
Input database alias   = SAMPLE
Locks held              = 5
Applications currently connected = 1
Agents currently waiting on locks = 0
Snapshot timestamp     = 06-05-2002 17:08:25.048027

```

```

Application handle     = 8
Application ID         = *LOCAL.DB2.0098C5210749
Sequence number       = 0001
Application name       = db2bp.exe
CONNECT Authorization ID = DB2ADMIN
Application status     = UOW Waiting
Status change time    = Not Collected
Application code page  = 1252
Locks held            = 5
Total wait time (ms)  = 0

```

List Of Locks

```

Lock Name              = 0x02000300050000000000000052
Lock Attributes        = 0x00000000
Release Flags         = 0x00000001
Lock Count             = 1
Hold Count             = 0
Lock Object Name      = 5
Object Type           = Row
Tablespace Name       = USERSPACE1
Table Schema          = DB2ADMIN
Table Name            = STAFF
Mode                  = U

```

```

Lock Name              = 0x02000300000000000000000054
Lock Attributes        = 0x00000000
Release Flags         = 0x00000001
Lock Count             = 1
Hold Count             = 0
Lock Object Name      = 3
Object Type           = Table
Tablespace Name       = USERSPACE1
Table Schema          = DB2ADMIN
Table Name            = STAFF
Mode                  = IX

```

```

Lock Name              = 0x01000000010000000100810056
Lock Attributes        = 0x00000000
Release Flags         = 0x40000000
Lock Count             = 1
Hold Count             = 0
Lock Object Name      = 0
Object Type           = Internal Variation Lock
Mode                  = S

```

```

Lock Name              = 0x41414141414A48520000000041
Lock Attributes        = 0x00000000

```

```

Release Flags          = 0x40000000
Lock Count             = 1
Hold Count             = 0
Lock Object Name      = 0
Object Type            = Internal Plan Lock
Mode                   = S

Lock Name              = 0x434F4E544F4B4E310000000041
Lock Attributes        = 0x00000000
Release Flags          = 0x40000000
Lock Count             = 1
Hold Count             = 0
Lock Object Name      = 0
Object Type            = Internal Plan Lock
Mode                   = S

```

From this snapshot, you can see that there is currently one application connected to the SAMPLE database, and it is holding five locks.

```

Locks held              = 5
Applications currently connected = 1

```

Note that the time (Status change time) when the Application status became UOW Waiting is returned as Not Collected. This is because the UOW switch is OFF.

The lock snapshot also returns the total time spent so far in waiting for locks, by applications connected to this database.

```

Total wait time (ms)    = 0

```

---

## Subsection snapshots

On systems that use inter-partition parallelism, the SQL compiler partitions the access plan for an SQL statement into subsections. Each subsection is executed by a different DB2 agent (or agents for SMP).

The access plan for an SQL statement generated by the DB2 code generator during compilation can be obtained using the db2expln command. As an example, selecting all the rows from a table that is partitioned across several partitions might result in an access plan having two subsections:

1. Subsection 0, the coordinator subsection, whose role is to collect rows fetched by the other DB2 agents (subagents) and return them to the application.
2. Subsection 1, whose role is to perform a table scan and return the rows to the coordinating agent.

In this simple example, subsection 1 would be distributed across all the database partitions. There would be a subagent executing this subsection on each physical partition of the database partition group to which this table belongs.

The database system monitor allows you to correlate run-time information with the access plan, which is compile-time information. With inter-partition parallelism, the monitor breaks information down to the subsection level. For example, when the statement monitor switch is ON, a GET SNAPSHOT FOR APPLICATION will return information for each subsection executing on this partition, as well as totals for the statement.

The subsection information returned for an application snapshot includes:

- the number of table rows read/written
- CPU consumption

- elapsed time
- the number of table queue rows sent and received from other agents working on this statement. This allows you to track the execution of a long running query by taking a series of snapshots.
- subsection status. If the subsection is in a WAIT state, because it is waiting for another agent to send or receive data, then the information also identifies the partition or partitions preventing the subsection from progressing in its execution. You may then take a snapshot on these partitions to investigate the situation.

The information logged by a statement event monitor for each subsection after it has finished executing includes: CPU consumption, total execution, time, and several other counters.

---

## Global snapshots on partitioned database systems

On a partitioned database system, you can use the snapshot monitor to take a snapshot of the current partition, a specified partition, or all partitions. When taking a global snapshot across all the partitions of a partitioned database, data is aggregated before the results are returned.

Data is aggregated for the different element types as follows:

- **Counters, Time, and Gauges**  
Contains the sum of all like values collected from each partition in the instance. For example, `GET SNAPSHOT FOR DATABASE XYZ ON TEST GLOBAL` would return the number of rows read (`rows_read`) from the database for all partitions in the partitioned database instance.
- **Watermarks**  
Returns the highest (for high water) or lowest (for low water) value found for any partition in the partitioned database system. If the value returned is of concern, then snapshots for individual partitions can be taken to determine if a particular partition is over utilized, or if the problem is instance-wide.
- **Timestamp**  
Set to the timestamp value for the partition where the snapshot monitor instance agent is attached. Note that all timestamp values are under control of the `timestamp` monitor switch.
- **Information**  
Returns the most significant information for a partition that may be impeding work. For example, for the element `app1_status`, if the status on one partition was UOW Executing, and on another partition Lock Wait, Lock Wait would be returned, since it is the state that's holding up execution of the application.

You can also reset counters, set monitor switches, and retrieve monitor switch settings for individual partitions or all partitions in your partitioned database.

**Note:** When taking a global snapshot, if one or more partitions encounter an error, then data is collected from the partitions where the snapshot was successful and a warning (sqlcode 1629) is also returned. If a global get or update of monitor switches, or a counter reset fails on one or more partitions, then those partitions will not have their switches set, or data reset.

## Snapshot monitor self-describing data stream

After you capture a snapshot with the db2GetSnapshot API, the API returns the snapshot output as a self-describing data stream. Figure 2 shows the structure of the data stream and Table 43 on page 104 provides some examples of the logical data groups and monitor elements that might be returned.

**Note:** Descriptive names are used for the identifiers in the examples and tables. These names are prefixed by **SQLM\_ELM\_** in the actual data stream. For example, collected would appear as **SQLM\_ELM\_COLLECTED** in the snapshot monitor output. Types are prefixed with **SQLM\_TYPE\_** in the actual data stream. For example, headers appear as **SQLM\_TYPE\_HEADER** in the data stream.

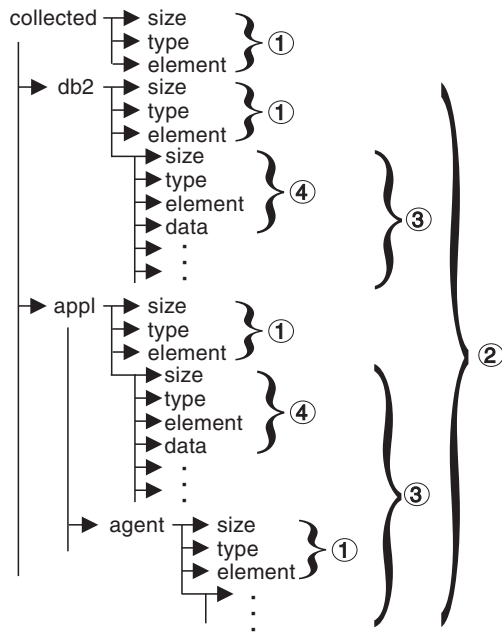


Figure 2. Snapshot Monitor Data Stream

1. Each logical data group begins with a header that indicates its size and name. This size does not include the volume of data taken up by the header itself.
2. Size in the collected header returns the total size of the snapshot.
3. The size element in other headers indicates the size of all the data in that logical data group, including any subordinate groupings.
4. Monitor element information follows its logical data group header and is also self-describing.

Table 43. Sample Snapshot Data Stream

| Logical Data Group | Data Stream         | Description                                      |
|--------------------|---------------------|--------------------------------------------------|
| collected          | 1000                | Size of snapshot data (in bytes).                |
|                    | header              | Indicates the start of a logical data group.     |
|                    | collected           | Name of the logical data group.                  |
|                    | 4                   | Size of the data stored in this monitor element. |
|                    | u32bit              | Monitor element type - unsigned 32 bit numeric.  |
|                    | server_db2_type     | The name of the monitor element collected.       |
|                    | sqlf_nt_server      | The collected value for this element.            |
|                    | 2                   | Size of the data stored in this monitor element. |
|                    | u16bit              | Monitor element type - unsigned 16 bit numeric.  |
|                    | node_number         | The name of the monitor element collected.       |
|                    | 3                   | The collected value for this element.            |
|                    | db2                 | 200                                              |
| header             |                     | Indicates the start of a logical data group.     |
| db2                |                     | Name of the logical data group.                  |
|                    | 4                   | Size of the data stored in this monitor element. |
|                    | u32bit              | Monitor element type - unsigned 32 bit numeric.  |
|                    | sort_heap_allocated | The name of the monitor element collected.       |
|                    | 16                  | The collected value for this element.            |
|                    | 4                   | Size of the data stored in this monitor element. |
|                    | u32bit              | Monitor element type - unsigned 32 bit numeric.  |
|                    | local_cons          | The name of the monitor element collected.       |
|                    | 3                   | The collected value for this element.            |
|                    | ...                 | ...                                              |
| appl               | 100                 | Size of the appl element data in the snapshot.   |
|                    | header              | Indicates the start of a logical data group.     |
|                    | appl                | Name of the logical data group.                  |
|                    | 4                   | Size of the data stored in this monitor element. |
|                    | u32bit              | Monitor element type - unsigned 32 bit numeric.  |
|                    | locks_held          | The name of the monitor element collected.       |
|                    | 3                   | The collected value for this element.            |
|                    | ...                 | ...                                              |

Table 43. Sample Snapshot Data Stream (continued)

| Logical Data Group | Data Stream | Description                                      |
|--------------------|-------------|--------------------------------------------------|
| agent              | 50          | Size of the agent portion of the appl structure. |
|                    | header      | Indicates the start of a logical data group.     |
|                    | agent       | Name of the logical data group.                  |
|                    | 4           | Size of the data stored in this monitor element. |
|                    | u32bit      | Monitor element type - 32 bit numeric.           |
|                    | agent_pid   | The name of the monitor element collected.       |
|                    | 12          | The collected value for this element.            |
| ...                | ...         | ...                                              |

The db2GetSnapshot() routine returns the self-describing snapshot data in the user-supplied buffer. Data is returned in the logical data groupings associated with the type of snapshot being captured.

Each item returned by a snapshot request contains fields that specify its size and type. The size can be used to parse through the returned data. A field's size can also be used to skip over a logical data group. For example, to skip over the DB2 record you need to determine the number of bytes in the data stream. Use the following formula to calculate the number of bytes to skip:

size of the db2 logical data grouping + sizeof(sqlm\_header\_info)

## Monitoring with db2top in interactive mode commands

The db2top monitoring utility quickly and efficiently monitors a complex DB2 environment. It combines DB2 snapshot information from all database partitions and provides a dynamic, real-time view of a running DB2 system using a text-based user interface.

When you run db2top in interactive mode, you can issue the following commands:

- A** Monitor either the primary or the secondary database in a HADR cluster.
- a** Goto application details for agent (or restrict on agent on statement screen). The db2top command will prompt for the agent-id.
- B** Display the main consumer of critical server resources (Bottleneck Analysis).
- c** This option allows to change the order of the columns displayed on the screen. The syntax is in the form: 1,2,3,... where 1,2,3 correspond respectively to the 1st, 2nd and 3rd columns displayed. These are the column numbers to use when specifying a sort criteria.

When you use the c switch key, a screen is shown specifying the order of the columns displayed on screen. The left part of the screen displays the default order and column numbers; the right part of the screen displays the current ordering. To change the order of the columns, enter the new column order in the text field at the bottom of the screen. Next, enter the relative column positions as displayed on the left, separated by commas. Not all columns need to be specified. This column ordering can be saved in \$DB2TOPRC for subsequent db2top monitoring sessions by selecting w.



You can sort and select in which order the columns are displayed on the screen. Valid keywords for column ordering in the .db2toprc file are:

- sessions=
- tables=
- tablespaces=
- bufferpools=
- dynsql=
- statements=
- locks=
- utilities=
- federation=

- b** Goto buffer pool screen.
- C** Toggle snapshot data collector on/off.
- d** Goto database screen.
- D** Goto the dynamic SQL screen.
- f** Freeze screen.
- F** Monitor federated queries on the primary server.
- G** Toggle graph on/off.
- h** Go to Help screen
- H** Goto the history screen
- i** Toggle idle sessions on/off.
- k** Toggle actual vs delta values.
- l** Goto sessions screen.
- L** Allows to display the complete query text from the SQL screen. Regular DB2 explain can then be run using e or X options.
- m** Display memory pools.
- o** Display session setup.
- p** Goto the partitions screen.
- P** Select db partition on which to issue snapshot.
- q** Quit db2top.
- R** Reset snapshot data.
- s** Goto the statements screen.
- S** Run native DB2 snapshot.
- t** Goto table spaces screen.
- T** Goto tables screen
- u** Display active utilities and aggregate them across database partitions.
- U** Goto the locks screen.
- V** Set default explains schema.
- w** Write session settings to .db2toprc.
- W** Watch mode for agent\_id, os\_user, db\_user, application or netname.

Statements returned by the session snapshot (option I) will be written to agent.sql, os\_user-agent.sql, db\_user-agent.sql, application-agent.sql or netname-agent.sql. When issued from the dynamic SQL screen (option D), statements will be written to db2adv.sql in a format compatible with db2advvis.

- X Toggle extended mode on/off.
- z|Z Sort on ascending or descending order.
- / Enter expression to filter data. Expression must conform to regular expression. Each function (screen) can be filtered differently. The regexp check is applied to the whole row.
- <l> Move to left or right of screen.

The following switches apply to the applications screen only:

- r Return to previous function.
- R Toggle automatic refresh.
- g Toggle graph on/off.
- X Toggle extended mode on/off.
- d Display agents.

To start db2top in interactive mode, issue the following command:

```
db2top -d <database name>
```

When you type  
db2top -d sample

the following output is displayed:

```
[\]11:57:10,refresh=2secs(0.000) Inactive,part=[1/1],<instanceName>:sample
[d=Y,a=N,e=N,p=ALL] [qp=off]

[/]: When rotating, it means that db2top is waiting between two snapshots, otherwise, it means db2top is waiting
      from an answer from DB2
11:57:10: current time
refresh=2secs: time interval
refresh=!secs: Exclamation mark means the time to process the snapshot by DB2 is longer than the refresh interval.
               In this case, db2top will increase the interval by 50%. If this occurs too often because the system is too busy,
               you can either increase the snapshot interval (option I), monitor a single database partition (option P), or turn
               off extended display mode (option x)
0.000 : time spent inside DB2 to process the snapshot
d=Y/N : delta or cumulative snapshot indicator (command option -k or option k).
a=Y/N : Active only or all objects indicator (-a command option set or i)
e=Y/N : Extended display indicator
p=ALL : All database partitions
p=CUR: Current database partition (-P command option with no partition number specified)
p=3 : target database partition number: say 3

Inactive: : Shows inactive if DB2 is not running, otherwise displays the platform on which DB2 is running
part=[1/1] : active database partition number vs total database partition number. For example, part=[2,3] means one
             database partition out of 3 is down (2 active, 3 total)
<instanceName> : instance name
sample : database name
qp=off/on : query patroller indicator (DYNMGMT database configuration parameter) for the database partition
           on which db2top is attached
```

The following example demonstrates running the db2top monitoring utility in interactive mode in a partitioned database environment:

```
db2top -d TEST -n mynode -u user -p passwd -V skm4 -B -i 1
The command parameters are as follows:
-d TEST      # database name
-n mynode    # node name
```

```

-u user      # user id
-p passwd   # password
-V skm4     # Schema name
-B          # Bold enabled
-i 1        # Screen update interval: 1 second

```

## .db2toprc configuration file

The .db2toprc configuration file is a user generated file used to set parameters at initialization time for the db2top monitoring utility. The db2top utility will search for the location of the .db2toprc file using the user-defined variable *\$db2topRC*. If the variable is not set, db2top will first search for the .db2toprc file in the current directory and then in the home directory. The .db2toprc file is user generated.

### Environment variables

You can set the following environment variables:

- **DB2TOPRC**

A user defined environment variable that stores the location of the .db2toprc file. For example, on Linux, you can define DB2TOPRC as: `export db2topRC=~/.db2toprc`.

If the variable is not set by the user, db2top will first search for the .db2toprc file in the current directory and then in the home directory.

- **DB2DBDFT**

This variable specifies the database alias name of the database to be used for implicit connects. It is used when no database name is specified on the command line or in the .db2toprc configuration file.

- **EDITOR**

This system environment variable specifies the command used to start the text editor used to display the results of explain or native snapshots.

If this variable is not set, vi is used.

### Structure

Some of the entries in the .db2toprc file are described here.

#### **cpu=command**

Use this entry to display the results of CPU activity on the second line at the right of the screen output. For example:

```
cpu=vmstat 2 2 | tail -1 | awk '{printf("%d(usr+sys)0,$14+$15);}'
```

displays `Cpu=2(usr+sys)` on the right of the screen.

#### **io=command**

Use this entry to specify a command and display the result on the second line at the left of the screen output. For example:

```
io=vmstat 2 2 | tail -1 | awk '{printf("%d(bi+bo)0,$10+$11);}'
```

displays `Disk=76(bi+bo)` on the left of the screen.

Both commands run as background processes and the fields on the screen are updated asynchronously.

#### **shell alias=command**

Use this shell entry to specify a user defined command, for example: `shell M=top` spawns top from a db2top session when entering M. It returns to the current screen upon exit.

### function alias=command

Use this entry to specify a user defined command, for example: function N=netstat creates a new function called N that repeatedly displays the output of netstat. There can be multiple function entries. They must be placed on separate lines. For example:

```
function Q=netstat
function N=df -k
```

### sort=command

Use this entry to specify a sort order, for example: sort=command creates a default sort order for this function, where command is the column number. It can be either ascending or descending. Sort is valid for sessions, tables, tablespace, bufferpool, dynsql, statements, locks, utilities and federation.

## Sample .db2toprc file

There is no default .db2toprc configuration file. However, you can press "W" to create a .db2toprc for the current setup. Use the following sample .db2toprc file as a reference. Comments have been added to all entries.

```
# db2top configuration file
# On UNIX, should be located in $HOME/<cmdname> .db2toprc</cmdname>
# File generated by db2top-1.0a
#
node= # [-n] nodename
database=sample # [-d] databasename
user= # [-u] database user
password= # [-p] user password (crypted)
schema= # [-V] default schema for explains
interval=2 # [-i] sampling interval
active=OFF # [-a] display active sessions only (on/off)
reset=OFF # [-R] Reset snapshot at startup (on/off)
delta=ON # [-k] Toggle display of delta/cumulative values (on/off)
gauge=ON # display graph on sessions list (on/off)
colors=ON # True if terminal supports colors. Informs GE_WRS if it can display information with colors
graphic=ON # True if terminal supports semi graphical characters (on/off).
port= # Port for network collection
streamsize=size # Max collection size per hour (eg. 1024 or 1K : K, M or G)
# Command to get cpu usage information from OS
cpu=vmstat 2 2 | tail -1 | awk '{printf("%d(usr+sys)0,$14+$15);}'
# Command to get IO usage information from OS
io=vmstat 2 2 | tail -1 | awk '{printf("%d(bi+bo)0,$10+$11);}'
# Ordering of information in sessions screen
# Column order for the session screen (option l)
sessions=0,1,18,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,19,20,21,22,23
# Column order for the tables screen (option T)
tables=0,1,2,4,3,5,6,7
# Column order for the tablespaces screen (option t). The display will be sorted in ascending order on column #22
tablespaces=0,1,18,2,3,4,5,6,7,8, sort=22a
# Column order for the bufferpool screen (option b)
bufferpools=0,1,18,2,3,4,5,6,7,8,9,10
# Column order for the Dynamic SQL screen (option D)
dynsql=0,1,18,2,3,4,5,6,7,8,9
statements=0,1
locks=0,1
utilities=0 # contains the default column and sort order for the utility screen
federation=0,2,4 # contains the default column and sort order for the federation screen

# User defined commands
shell P=top
function N=date && netstat -t tcp
```



---

## Chapter 5. Switch-based monitoring concepts

---

### System monitor switches

System monitor switches control how snapshot monitors and some event monitors collect data.

**Note:** These system monitor switches do not affect the unit of work event monitor and locking event monitor, which were introduced in DB2 Version 9.7.

The snapshot monitor and some event monitors report data collected by the system monitor. Collecting system monitor data introduces processing overhead for the database manager. For example, in order to calculate the execution time of SQL statements, the database manager must make calls to the operating system to obtain timestamps before and after the execution of every statement. These types of system calls are generally expensive. Another form of overhead incurred by the system monitor is increased memory consumption. For every monitor element tracked by the system monitor, the database manager uses its memory to store the collected data.

In order to minimize the overhead involved in maintaining monitoring information, monitor switches control the collection of potentially expensive data by the database manager. Each switch has only two settings: ON or OFF. If a monitor switch is OFF, the monitor elements under that switch's control do not collect any information. There is a considerable amount of basic monitoring data that is not under switch control, and will always be collected regardless of switch settings.

Each monitoring application has its own logical view of the monitor switches (and the system monitor data). Upon startup each application inherits its monitor switch settings from the `dft_monswitches` parameters in the database manager configuration file (at the instance level). A monitoring application can alter its monitor switch settings with the `UPDATE MONITOR SWITCHES USING MONSWITCH OFF/ON` command. The `MONSWITCH` parameter holds values found in the Monitor Switch column in the Snapshot Monitor Switches table below. Changes to the switch settings at the application level only affect the application from where the switch was changed.

Instance-level monitor switches can be changed without stopping the database management system. To do this use the `UPDATE DBM CFG USING DBMSWITCH OFF/ON` command. The `DBMSWITCH` parameter holds values from the DBM Parameter column in the Snapshot Monitor Switches table below. This dynamic updating of switches requires that the application performing the update be explicitly attached to the instance for the updates to dynamically take effect. Other existing snapshot applications will not be affected by a dynamic update. New monitoring applications will inherit the updated instance-level monitor switch settings. For an existing monitoring application to inherit the new default monitor switch values, it must terminate and re-establish its attachment. Updating the switches in the database manager configuration file will update the switches for all partitions in a partitioned database.

The database manager keeps track of all the snapshot monitoring applications and their switch settings. If a switch is set to ON in one application's configuration,

then the database manager always collects that monitor data. If the same switch is then set to OFF in the application's configuration, then the database manager will still collect data as long as there is at least one application with this switch turned ON.

The collection of time and timestamp elements is controlled by the **TIMESTAMP** switch. Turning this switch OFF (it is ON by default) instructs the database manager to skip any timestamp operating system calls when determining time or timestamp-related monitor elements. Turning this switch OFF becomes important as CPU utilization approaches 100%. When this occurs, the performance degradation caused by issuing timestamps increases dramatically. For monitor elements that can be controlled by the **TIMESTAMP** switch and another switch, if either of the switches is turned OFF, data is not collected. Therefore, if the **TIMESTAMP** switch is turned OFF, the overall cost of data under the control of other monitor switches is greatly reduced.

Event monitors are not affected by monitor switches in the same way as snapshot monitoring applications. When an event monitor is defined, it automatically turns ON the instance level monitor switches required by the specified event types. For example, a deadlock event monitor will automatically turn ON the **LOCK** monitor switch. The required monitor switches are turned ON when the event monitor is activated. When the event monitor is deactivated, the monitor switches are turned OFF.

The **TIMESTAMP** monitor switch is not set automatically by event monitors. It is the only monitor switch that controls the collection of any monitor elements belonging to event monitor logical data groupings. If the **TIMESTAMP** switch is OFF, most of the timestamp and time monitor elements collected by event monitors will not be collected. These elements are still written to the specified table, file, or pipe, but with a value of zero.

*Table 44. Snapshot Monitor Switches*

| Monitor Switch | DBM Parameter     | Information Provided                      |
|----------------|-------------------|-------------------------------------------|
| BUFFERPOOL     | DFT_MON_BUFPOOL   | Number of reads and writes, time taken    |
| LOCK           | DFT_MON_LOCK      | Lock wait times, deadlocks                |
| SORT           | DFT_MON_SORT      | Number of heaps used, sort performance    |
| STATEMENT      | DFT_MON_STMT      | Start/stop time, statement identification |
| TABLE          | DFT_MON_TABLE     | Measure of activity (rows read/written)   |
| UOW            | DFT_MON_UOW       | Start/end times, completion status        |
| TIMESTAMP      | DFT_MON_TIMESTAMP | Timestamps                                |

Before capturing a snapshot or using an event monitor, you must determine what data you need the database manager to gather. If you want any of the following special types of data to be collected in a snapshot, you will need to set the appropriate monitor switches.

- Buffer pool activity information
- Lock, lock wait, and time related lock information

- Sorting information
- SQL statement information
- Table activity information
- Times and timestamp information
- Unit of work information

The switches corresponding to the above information types are all OFF by default, except for the switch corresponding to times and timestamp information, which is ON by default.

Event monitors are only affected by the time and timestamp information switch. All other switch settings have no effect on the data collected by event monitors.

## Setting system monitor switches from the CLP

System monitor switches control the collection of data by the system monitor. By setting certain monitor switches to ON, you can collect specific types of monitor data.

The application performing any monitor switch updates must have an instance attachment. You must have one of SYSADM, SYSCTRL, SYSMAINT, or SYSMON authority to use the following commands:

- UPDATE MONITOR SWITCHES
- GET MONITOR SWITCHES
- GET DATABASE MANAGER MONITOR SWITCHES

You must have SYSADM authority to use the UPDATE DBM CFG command.

- To activate any of the local monitor switches, use the UPDATE MONITOR SWITCHES command. The switches will remain active until the application (CLP) detaches, or until they are deactivated with another UPDATE MONITOR SWITCHES command. The following example updates all of the local monitor switches to be ON:

```
db2 update monitor switches using BUFFERPOOL on LOCK on
      SORT on STATEMENT on TIMESTAMP on TABLE on UOW on
```

- To deactivate any of the local monitor switches, use the UPDATE MONITOR SWITCHES command. The following example updates all of the local monitor switches to be OFF:

```
db2 update monitor switches using BUFFERPOOL off, LOCK off,
      SORT off, STATEMENT off, TIMESTAMP off, TABLE off, UOW off
```

The following is an example of the output you would expect to see after issuing the above UPDATE MONITOR SWITCH command:

### Monitor Recording Switches

```
Switch list for db partition number 1
Buffer Pool Activity Information (BUFFERPOOL) = OFF
Lock Information (LOCK) = OFF
Sorting Information (SORT) = OFF
SQL Statement Information (STATEMENT) = OFF
Table Activity Information (TABLE) = OFF
Unit of Work Information (UOW) = OFF
Get timestamp information (TIMESTAMP) = OFF
```

- It is also possible to manipulate the monitor switches at the database manager level. This involves changing the dft\_monswitches parameters in the database



manager configuration file, using the UPDATE DBM CFG command. In the following example, only lock switch controlled information is to be collected in addition to the basic information.

```
db2 update dbm cfg using DFT_MON_LOCK on
```

Whenever a monitoring application is started, it inherits its monitor switch settings from the database manager. Any changes to the database manager's monitor switch settings will not impact any running monitoring applications. Monitoring applications must reattach themselves to the instance to pick up any changes to monitor switch settings.

- For partitioned database systems, you can set monitor switches specifically for a certain partition, or globally for all partitions.

1. To set a monitor switch (for example, BUFFERPOOL) for a specific partition (for example, partition number 3), issue the following command:

```
db2 update monitor switches using BUFFERPOOL on
    at dbpartitionnum 3
```

2. To set a monitor switch (for example, SORT) for all partitions, issue the following command:

```
db2 update monitor switches using SORT on global
```

- To check the status of the local monitor switches use the GET MONITOR SWITCHES command.

```
db2 get monitor switches
```

- For partitioned database systems, you can view the monitor switch settings specifically for a certain partition, or globally for all partitions.

1. To view the monitor switch settings for a specific partition (for example, partition number 2), issue the following command:

```
db2 get monitor switches at dbpartitionnum 2
```

2. To view the monitor switch settings for all partitions, issue the following command:

```
db2 get monitor switches global
```

- To check the status of the monitor switches at the database manager level (or instance level) use the GET DATABASE MANAGER MONITOR SWITCHES command. This command will show the overall switch settings for the instance being monitored.

```
db2 get database manager monitor switches
```

The following is an example of the output you should expect to see after issuing the above command:

#### DBM System Monitor Information Collected

```
Switch list for db partition number 1
Buffer Pool Activity Information (BUFFERPOOL) = OFF
Lock Information (LOCK) = ON 10-25-2001 16:04:39
Sorting Information (SORT) = OFF
SQL Statement Information (STATEMENT) = OFF
Table Activity Information (TABLE) = OFF
Unit of Work Information (UOW) = OFF
Get timestamp information (TIMESTAMP) = OFF
```

Now that you have set the desired monitor switches and confirmed the switch settings, you are ready to capture and collect monitor data.

## Setting system monitor switches from a client application

System monitor switches control the collection of data by the system monitor. By setting certain monitor switches to ON, you can collect specific types of monitor data.

The application performing any monitor switch updates must have an instance attachment. You must have SYSADM, SYSCTRL, SYSMAINT, or SYSMON authority to use the db2MonitorSwitches API.

1. Include the following DB2 libraries: sqlutil.h and db2ApiDf.h. These are found in the include subdirectory under sqllib.

```
#include <sqlutil.h>
#include <db2ApiDf.h>
#include <string.h>
#include <sqlmon.h>
```

2. Set switch lists buffer unit size to 1 KB.

```
#define SWITCHES_BUFFER_UNIT_SZ 1024
```

3. Initialize the sqlca, db2MonitorSwitches, and sqlm\_recording\_group structures. Also, initialize a pointer to contain the switch lists buffer, and establish the buffer's size.

```
struct sqlca sqlca;
memset (&sqlca, '\0', sizeof(struct sqlca));
db2MonitorSwitchesData switchesData;
memset (&switchesData, '\0', sizeof(switchesData));
struct sqlm_recording_group switchesList[SQLM_NUM_GROUPS];
memset(switchesList, '\0', sizeof(switchesList));
sqluint32 outputFormat;
static sqluint32 switchesBufferSize = SWITCHES_BUFFER_UNIT_SZ;
char *switchesBuffer;
```

4. Initialize the buffer, which is to hold the switch list output.

```
switchesBuffer = (char *)malloc(switchesBufferSize);
memset(switchesBuffer, '\0', switchesBufferSize);
```

5. To alter the state of the local monitor switches, alter the elements in the sqlm\_recording\_group structure (named switchesList as indicated in the previous step). For a monitor switch to be turned on, the parameter input\_state is to be set to SQLM\_ON. For a monitor switch to be turned off, the parameter input\_state must be set to SQLM\_OFF.

```
switchesList[SQLM_UOW_SW].input_state = SQLM_ON;
switchesList[SQLM_STATEMENT_SW].input_state = SQLM_ON;
switchesList[SQLM_TABLE_SW].input_state = SQLM_ON;
switchesList[SQLM_BUFFER_POOL_SW].input_state = SQLM_OFF;
switchesList[SQLM_LOCK_SW].input_state = SQLM_OFF;
switchesList[SQLM_SORT_SW].input_state = SQLM_OFF;
switchesList[SQLM_TIMESTAMP_SW].input_state = SQLM_OFF;
switchesData.piGroupStates = switchesList;
switchesData.poBuffer = switchesBuffer;
switchesData.iVersion = SQLM_DBMON_VERSION9_5;
switchesData.iBufferSize = switchesBufferSize;
switchesData.iReturnData = 0;
switchesData.iNodeNumber = SQLM_CURRENT_NODE;
switchesData.poOutputFormat = &outputFormat;
```

**Note:** SQLM\_TIMESTAMP\_SW is unavailable if iVersion is less than SQLM\_DBMON\_VERSION8.

6. To submit the changes to switch settings, call the db2MonitorSwitches() function. Pass the db2MonitorSwitchesData structure (named switchesData in this example) as a parameter to the db2MonitorSwitches API. The switchesData contains the sqlm\_recording\_group structure as a parameter.

```
db2MonitorSwitches(db2Version810, &switchesData, &sqlca);
```

7. Process the switch list data stream from the switch list buffer.

8. Clear the switch list buffer.

```
free(switchesBuffer);
free(pRequestedDataGroups);
```

Now that you have set the desired monitor switches and confirmed the switch settings, you are ready to capture and collect monitor data.

## System monitor switches self-describing data stream

After you update or view the current system monitor switch settings with the `db2MonitorSwitches` API, the API returns the switch settings as a self-describing data stream. Figure 3 shows the structure of the switch list information that may be returned for a partitioned database environment.

### Note:

1. Descriptive names are used for the identifiers in the examples and tables. These names are prefixed by `SQLM_ELM_` in the actual data stream. For example, `db_event` would appear as `SQLM_ELM_DB_EVENT` in the event monitor output. Types are prefixed with `SQLM_TYPE_` in the actual data stream. For example, headers appear as `SQLM_TYPE_HEADER` in the data stream.
2. For global switch requests the partition order of the returned information can be different in each switch request. In this case, a partition id is included in the data stream.

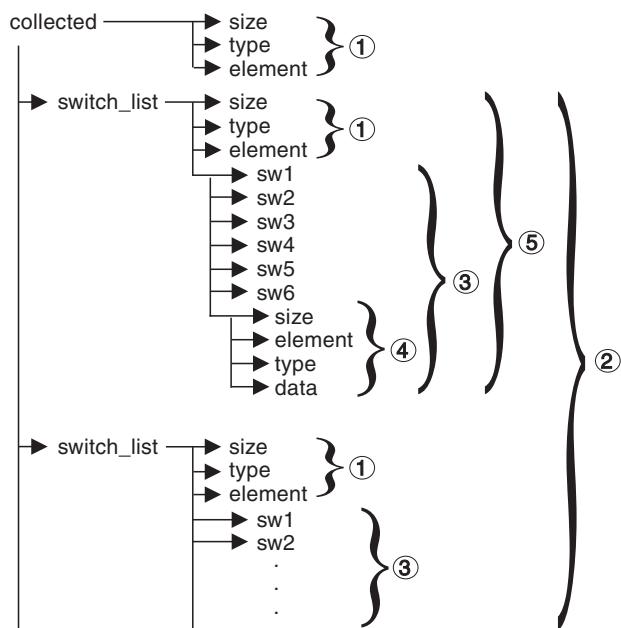


Figure 3. Switch List Monitor Data Stream

1. Each logical data group begins with a header that indicates its size and name. This size does not include the volume of data taken up by the header itself.
2. Size in the collected header returns the total size of all monitor switch lists for all partitions.
3. The size element in switch list header indicates the size of switch data for that partition.

4. Switch information is self-describing.
5. For a non-partitioned database, the switch settings for the stand alone partition are returned. That is, only one switch list is returned.

---

## Database system monitor data organization

The system monitor collects and stores information that you can access using interfaces to the snapshot monitor and some event monitors. The database system monitor stores information it collects in entities called *monitor elements* (these were previously known as data elements). Each monitor element stores information regarding one specific aspect of the state of the database system. In addition, monitor elements are identified by unique names and store a certain type of information.

The following are the available element types used by the system monitor in which monitor elements store data:

### Counter

Counts the number of times an activity occurs. Counter values increase during monitoring. Most counter elements can be reset.

**Gauge** Indicates the current value for an item. Gauge values can go up and down depending on database activity (for example, the number of locks held). Gauge elements can not be reset.

### Watermark

Indicates the highest (maximum) or lowest (minimum) value an element has reached since monitoring was started. Watermark elements can not be reset.

### Information

Provides reference-type details of your monitoring activities. This can include items such as partition names, aliases, and path details. Information elements can not be reset.

### Timestamp

Indicates the date and time that an activity took place by providing the number of seconds and microseconds that have elapsed since January 1, 1970. For the snapshot monitor and event monitors, the collection of timestamp elements is controlled by the `TIMESTAMP` monitor switch. While this switch is on by default, you should turn it off for performance reasons if CPU utilization on the database instance approaches 100%. Timestamp elements can not be reset.

A value of 0 for the timestamp element means "not available". If you attempt to import this data, such a value will generate an out of range error (SQL0181). To avoid this error, update the value to any valid timestamp value before exporting the data.

**Time** Returns the number of seconds and microseconds spent on an activity. For the snapshot monitor and event monitors, the collection of most time elements is controlled by the `TIMESTAMP` monitor switch. While this switch is on by default, you should turn it off for performance reasons if CPU utilization on the database instance approaches 100%. Some time elements can be reset.

Monitor elements collect data for one or more logical data groups. A logical data group is a collection of monitor elements that gather database system monitoring information for a specific scope of database activity. Monitor elements are sorted in

logical data groups based on the levels of information they provide. For example, while snapshot monitoring, the Total Sort Time monitor element returns database (dbase), application (appl), and statement (stmt) information; hence, it appears in each of the logical data groups listed in parentheses.

Although many monitor elements are used by both the snapshot monitor and event monitors, they each use a distinct set of logical data groups. This is because the scopes of database activity for which you can capture a snapshot differ from those for which you can collect event data. Practically speaking, the overall set of monitor elements accessible from the snapshot monitor is different from those accessible from event monitors.

---

## Counter status and visibility

Among the monitor elements collected by the system monitor are several accumulating counters. These counters are incremented during the operation of the database or database manager, for example, every time an application commits a transaction.

Counters are initialized when their applicable object becomes active. For instance, the number of buffer pool pages read for a database (a basic monitor element) is set to zero when the database is activated.

Some counters that can be collected by the system monitor are controlled by monitor switches. If a particular monitor switch is off, the monitor elements under its control do not collect data. When a monitor switch is turned on, all the associated counters are reset to zero.

Counters returned by event monitors are reset to zero when the event monitor is activated.

Event monitor counting represents a count since one of the following starting points:

- Event monitor startup, for database, table space, and tables.
- Event monitor startup, for existing connections.
- Application connection, for connections made after the monitor was started.
- Start of the next transaction (unit of work) or statement after the monitor was started.
- Occurrence of a deadlock after the monitor was started.

Each event monitor and any monitoring application (an application using the snapshot monitor APIs) has its own logical view of the system monitor data. This means that when counters are reset or initialized, it only affects the event monitor or application that reset or initialized them. Event monitor counters cannot be reset, except by turning the event monitor off, and then on again. An application taking snapshots can reset its view of the counters at any time by using the RESET MONITOR command.

If you start a statement event monitor after a statement starts, the monitor will start collecting information when the next SQL statement starts. As a result, the event monitor will not return information about statements that the database manager is executing when the monitor was started. This is also true for transaction information.

---

## System monitor output: the self-describing data stream

Aside from presenting system monitor data on screen or storing it in SQL tables, you can develop a client application to process it. The system monitor returns monitor data via a self-describing data stream for both the snapshot monitor and event monitor. In a snapshot monitoring application you can call the snapshot APIs to capture a snapshot and then directly process the data stream.

Processing event monitor data is different, in that the event data is sent to the application at the pace database events occur. For a pipe event monitor, the application waits for event data to arrive, and then processes it when it does. For a file event monitor, the application parses event files, thus processing event records in batches.

This self-describing data stream allows you to parse through the returned data one element at a time. This opens up numerous monitoring possibilities, including looking for information regarding a particular application or a specific database state.

The returned monitor data is in the following format:

- size** The size (in bytes) of the data stored in the monitor element or logical data grouping. In the case of a logical data grouping, this is the size of all data in the logical group. For example, the database logical grouping (*db*) contains individual monitor elements (such as *total\_log\_used*) along with other logical data groupings, such as rollforward information (*rollforward*). This does not include the size taken up by the 'size', 'type', and 'element' information.
- type** The type of element stored in the data (for example, variable length string or signed 32 bit numeric value). An element type of *header* refers to a logical data grouping for an element.
- element id**  
The identifier for the monitor element that was captured by the monitor. In the case of a logical data grouping, this is the identifier for the group (for example, *collected*, *dbase*, or *event\_db*).
- data** The value collected by a monitor for a monitor element. In the case of a logical data grouping, the data is composed of the monitor elements belonging to it.

All timestamps in monitor elements are returned in two unsigned 4 byte monitor elements (seconds and microseconds). These represent the number of seconds since January 1, 1970 in GMT time.

The size element of strings in monitor elements represents the actual size of data for the string element. This size does not include a null terminator, as the strings are not null terminated.

---

## Memory requirements for monitor data

The memory required for monitor data is allocated from the monitor heap. Monitor heap size is controlled by the **mon\_heap\_sz** database configuration parameter. This parameter has a default value of *AUTOMATIC*, meaning that the monitor heap can increase as needed until the *instance\_memory* limit is reached.

If you configure the `mon_heap_sz` parameter manually, consider the following factors:

- The number of monitoring applications
- The number and nature of event monitors
- The monitor switches set
- The level of database activity

Consider increasing the value for the `mon_heap_sz` parameter if monitor commands fail with an SQLCODE of -973.

The following formula provides an approximation of the number of pages required for the monitor heap:

$$\begin{array}{r} \text{(Memory used by applications} \\ \text{Memory used by event monitors} \\ \text{Memory used by monitoring applications} \\ \text{Memory used by Gateway applications)} \end{array} \begin{array}{r} + \\ + \\ + \\ \end{array} / 4096$$

### Memory used by each application

- If the STATEMENT switch is off, zero
- If the STATEMENT switch is on:
  - Add 400 bytes for each statement being run at the same time. (That is, the number of open cursors that an application might have). This is *not* the cumulative total of statements an application has run.
  - If a partitioned database, add the following for each statement:
    - 200 bytes \* (average # of subsections)
- If the application has issued `sqleseti()` info, add the sizes of the `userid`, `aplname`, `workstation name` and `accounting string`.

### Memory used by each event monitor

For each event monitor of type ACTIVITIES:

- 3500 bytes
- If the event monitor is for type TABLES, add  $36K * (\text{number of CPU cores} + 1)$
- If the event monitor is for type FILE or PIPE, add  $2K * (\text{number of CPU cores} + 1)$

If you expect a heavy volume, add 250 megabytes for event records. Otherwise add a fraction that depends on the expected amount of work.

For each event monitor of type LOCKING or UOW:

- 3500 bytes
- $3K * (\text{number of CPU cores} + 1)$

If you expect a heavy volume, add 250 megabytes for event records. Otherwise add a fraction that depends on the expected amount of work.

For each event monitor of the following type: DATABASE, TABLES, TABLESPACES, BUFFERPOOLS, CONNECTIONS, DEADLOCK:

- 4100 bytes
- $2 * \text{BUFFERSIZE}$
- If the event monitor is written to a file, add 550 bytes.
- If the event monitor is for type DATABASE:
  - add 6000 bytes



- add 100 bytes for each statement in the statement cache
- If the event monitor is for type TABLES:
  - add 1500 bytes
  - add 70 bytes for each table accessed
- If the event monitor is for type TABLESPACES:
  - add 450 bytes
  - add 350 bytes for each table space
- If the event monitor is for type BUFFERPOOLS:
  - add 450 bytes
  - add 340 bytes for each buffer pool
- If the event monitor is for type CONNECTIONS:
  - add 1500 bytes
  - for each connected application:
    - add 750 bytes
  - remember to add the value from “Memory used by each application” on page 120.
- If an event monitor is of type DEADLOCK:
  - and the WITH DETAILS HISTORY is running:
    - add  $X*475$  bytes times the maximum number of concurrent applications you expect to be running, where X is the expected maximum number of statements in your application’s unit of work.
  - and the WITH DETAILS HISTORY VALUES is running:
    - also add  $X*Y$  bytes times the maximum number of concurrent applications you expect to be running, where Y is the expected maximum size of parameter values being bound into your SQL statements.

### **Memory used by each monitoring application**

- 250 bytes
- For each database being reset:
  - 350 bytes
  - Add 200 bytes for each REMOTE database.
  - If the SORT switch is on, add 25 bytes.
  - If the LOCK switch is on, add 25 bytes.
  - If the TABLE switch is on:
    - add 600 bytes
    - add 75 bytes per table accessed
  - If the BUFFERPOOL switch is on:
    - add 300 bytes
    - add 250 bytes per table space accessed
    - add 250 bytes per buffer pool accessed
  - If the STATEMENT switch is on:
    - add 2100 bytes
    - add 100 bytes per statement
  - For each application connected to the database:
    - add 600 bytes
    - add 200 bytes for every REMOTE database the application is connected to



- if the SORT switch is on, add 25 bytes
- if the LOCK switch is on, add 25 bytes
- if the BUFFERPOOL switch is on, add 250 bytes
- For each DCS database being reset:
  - add 200 bytes for the database
  - add 200 bytes for each application connected to the database
  - if the STATEMENT switch is ON, Transmission level data must be reset:
    - for each database, add 200 bytes for each transmission level
    - for each application, add 200 bytes for each transmission level

### Memory used by gateway applications

- 250 bytes for each host database (even if all switches are off)
- 400 bytes for each application (even if all switches are off)
- If the STATEMENT switch is on:
  - For each application, add 200 bytes for each statement being run at the same time (That is, the number of open cursors that an application might have). This is NOT the cumulative total of statements an application has run.
  - Transmission level data must be accounted for:
    - for each database, add 200 bytes for each transmission level
    - for each application, add 200 bytes for each transmission level
- If the UOW switch is on:
  - add 50 bytes for each application
- For each application using a TMDB (for SYNCPOINT TWOPHASE activity):
  - add 20 bytes plus the size of the XID itself
- For any application that has issued sqleseti to set client name, app name, wkstn or accounting:
  - add 800 bytes plus the size of the accounting string itself

---

## Monitoring buffer pool activity

The database server reads and updates all data from a buffer pool. Data is copied from disk to a buffer pool as it is required by applications.

Pages are placed in a buffer pool:

- by the agent. This is synchronous I/O.
- by the I/O servers (prefetchers). This is asynchronous I/O.

Pages are written to disk from a buffer pool:

- by the agent, synchronously
- by page cleaners, asynchronously

If the server needs to read a page of data, and that page is already in the buffer pool, then the ability to access that page is much faster than if the page had to be read from disk. It is desirable to **hit** as many pages as possible in the buffer pool. Avoiding disk I/O is an important factor in database performance, therefore proper configuration of the buffer pools is one of the most important considerations for performance tuning.

The buffer pool hit ratio indicates the percentage of time that the database manager did not need to load a page from disk in order to service a page request because the page was already in the buffer pool. The greater the buffer pool hit ratio, the lower the frequency of disk I/O.

The overall buffer pool hit ratio can be calculated as follows:

$$1 - ((\text{pool\_data\_p\_reads} + \text{pool\_xda\_p\_reads} + \text{pool\_index\_p\_reads} + \text{pool\_temp\_data\_p\_reads} + \text{pool\_temp\_xda\_p\_reads} + \text{pool\_temp\_index\_p\_reads}) / (\text{pool\_data\_l\_reads} + \text{pool\_xda\_l\_reads} + \text{pool\_index\_l\_reads} + \text{pool\_temp\_data\_l\_reads} + \text{pool\_temp\_xda\_l\_reads} + \text{pool\_temp\_index\_l\_reads})) * 100\%$$

This calculation takes into account all of the pages (index and data) that are cached by the buffer pool.

You can also use the BP\_HITRATIO administrative view as a convenient method of monitoring the hit ratio for your buffer pools.

For a large database, increasing the buffer pool size may have minimal effect on the buffer pool hit ratio. Its number of data pages may be so large, that the statistical chances of a hit are not improved by increasing its size. Instead, you might find that tuning the index buffer pool hit ratio achieves the desired result. This can be achieved using two methods:

1. Split the data and indexes into two different buffer pools and tune them separately.
2. Use one buffer pool, but increase its size until the index hit ratio stops increasing. The index buffer pool hit ratio can be calculated as follows:

$$(1 - ((\text{pool\_index\_p\_reads}) / (\text{pool\_index\_l\_reads}))) * 100\%$$

The first method is often more effective, but because it requires indexes and data to reside in different table spaces, it may not be an option for existing databases. It also requires tuning two buffer pools instead of one, which can be a more difficult task, particularly when memory is constrained.

You should also consider the impact that prefetchers may be having on the hit ratio. Prefetchers read data pages into the buffer pool anticipating their need by an application (asynchronously). In most situations, these pages are read just before they are needed (the desired case). However, prefetchers can cause unnecessary I/O by reading pages into the buffer pool that will not be used. For example, an application starts reading through a table. This is detected and prefetching starts, but the application fills an application buffer and stops reading. Meanwhile, prefetching has been done for a number of additional pages. I/O has occurred for pages that will not be used and the buffer pool is partially taken up with those pages.

Page cleaners monitor the buffer pool and asynchronously write pages to disk. Their goals are:

- Ensure that agents will always find free pages in the buffer pool. If an agent does not find free pages in the buffer pool, it must clean them itself, and the associated application will have a poorer response.
- Speed database recovery, if a system crash occurs. The more pages that have been written to disk, the smaller the number of log file records that must be processed to recover the database.

Although dirty pages are written out to disk, the pages are not removed from the buffer pool right away, unless the space is needed to read in new pages.

**Note:** Buffer pool information is typically gathered at a table space level, but the facilities of the database system monitor can roll this information up to the buffer pool and database levels. Depending on your type of analysis, you may need to examine this data at any or all of these levels.

## Database system monitor interfaces

| Monitoring task                                 | API                |
|-------------------------------------------------|--------------------|
| Capturing a snapshot                            | db2GetSnapshot     |
| Converting the self-describing data stream      | db2ConvMonStream   |
| Displaying the database system monitor switches | db2MonitorSwitches |
| Estimating the size of a snapshot               | db2GetSnapshotSize |
| Get/update monitor switches                     | db2MonitorSwitches |
| Resetting monitor counters                      | db2ResetMonitor    |
| Updating the database system monitor switches   | db2MonitorSwitches |

| Monitoring task                                                          | CLP Command                           |
|--------------------------------------------------------------------------|---------------------------------------|
| Analyzing event monitor output with a GUI tool                           | db2eva                                |
| Capturing a snapshot                                                     | GET SNAPSHOT                          |
| Displaying the database manager monitor switches                         | GET DATABASE MANAGER MONITOR SWITCHES |
| Displaying the monitoring application's monitor switches                 | GET MONITOR SWITCHES                  |
| Formatting the event monitor trace                                       | db2evmon                              |
| Generating sample SQL for write-to-table CREATE EVENT MONITOR statements | db2evtbl                              |
| Listing the active databases                                             | LIST ACTIVE DATABASES                 |
| Listing the applications connected to a database                         | LIST APPLICATIONS                     |
| Listing the DCS applications                                             | LIST DCS APPLICATIONS                 |
| Resetting monitor counters                                               | RESET MONITOR                         |
| Updating the database system monitor switches                            | UPDATE MONITOR SWITCHES               |

| Monitoring task               | SQL Statement           |
|-------------------------------|-------------------------|
| Activating an event monitor   | SET EVENT MONITOR STATE |
| Creating an event monitor     | CREATE EVENT MONITOR    |
| Deactivating an event monitor | SET EVENT MONITOR STATE |
| Removing an event monitor     | DROP                    |
| Writing event monitor values  | FLUSH EVENT MONITOR     |

| <b>Monitoring task</b>                                                    | <b>SQL Function</b>                                                             |
|---------------------------------------------------------------------------|---------------------------------------------------------------------------------|
| Determining the state of an event monitor                                 | EVENT_MON_STATE scalar function                                                 |
| Getting a database manager level snapshot                                 | SNAPDBM administrative view and SNAP_GET_DBM_V95 table function                 |
| Getting the current monitor switch settings at the database manager level | SNAPSWITCHES administrative view and SNAP_GET_SWITCHES table function           |
| Getting a fast communication manager snapshot                             | SNAPFCM administrative view and SNAP_GET_FCM table function                     |
| Getting a fast communication manager snapshot for a given partition       | SNAPFCM_PART administrative view and SNAP_GET_FCM_PART table function           |
| Getting a database level snapshot                                         | SNAPDB administrative view and SNAP_GET_DB_V95 table function                   |
| Getting an application level snapshot                                     | SNAPAPPL administrative view and SNAP_GET_APPL_V95 table function               |
| Getting an application level snapshot                                     | SNAPAPPL_INFO administrative view and SNAP_GET_APPL_INFO_V95 table function     |
| Getting an application level snapshot for lock wait information           | SNAPLOCKWAIT administrative view and SNAP_GET_LOCKWAIT table function           |
| Getting an application level snapshot for statement information           | SNAPSTMT administrative view and SNAP_GET_STMT table function                   |
| Getting an application level snapshot for agent information               | SNAPAGENT administrative view and SNAP_GET_AGENT table function                 |
| Getting an application level snapshot for subsection information          | SNAPSUBSECTION administrative view and SNAP_GET_SUBSECTION table function       |
| Getting a buffer pool level snapshot                                      | SNAPBP administrative view and SNAP_GET_BP_V95 table function                   |
| Getting a table space level snapshot                                      | SNAPTbsp administrative view and SNAP_GET_TBSP_V91 table function               |
| Getting a table space level snapshot for configuration information        | SNAPTbsp_PART administrative view and SNAP_GET_TBSP_PART_V91 table function     |
| Getting a table space level snapshot for container information            | SNAPCONTAINER administrative view and SNAP_GET_CONTAINER_V91 table function     |
| Getting a table space level snapshot for quiescer information             | SNAPTbsp_QUIESCER administrative view and SNAP_GET_TBSP_QUIESCER table function |
| Getting a table space level snapshot for the ranges of a table space map  | SNAPTbsp_RANGE administrative view and SNAP_GET_TBSP_RANGE table function       |
| Getting a table level snapshot                                            | SNAPTAB administrative view and SNAP_GET_TAB_V91 table function                 |
| Getting a lock level snapshot                                             | SNAPLOCK administrative view and SNAP_GET_LOCK table function                   |
| Getting a snapshot of SQL statement cache information                     | SNAPDYN_SQL administrative view and SNAP_GET_DYN_SQL_V95 table function         |



---

## Chapter 6. Deprecated monitoring tools

---

### Health monitor

#### Monitoring database health

##### Introduction to the health monitor

The health monitor is a server-side tool that adds a management-by-exception capability by constantly monitoring the health of an instance and active databases. The health monitor also has the capability to alert a database administrator (DBA) of potential system health issues. The health monitor proactively detects issues that might lead to hardware failure, or to unacceptable system performance or capability. The proactive nature of the health monitor enables users to address an issue before it becomes a problem that affects system performance.

The health monitor checks the state of your system using health indicators to determine if an alert should be issued. Preconfigured actions can be taken in response to alerts. The health monitor can also log alerts in the administration notification log and send notifications by e-mail or pager. This management-by-exception model frees up valuable DBA resources by generating alerts to potential system health issues without requiring active monitoring.

The health monitor periodically gathers information about the health of the system with a very minimal impact to overall performance. It does not turn on any snapshot monitor switches to collect information.

##### Health indicators:

The health monitor uses health indicators to evaluate the health of specific aspects of database manager performance or database performance. A health indicator measures the health of some aspect of a particular class of database objects, such as table spaces. Criteria are applied to the measurement to determine healthiness. The criteria applied depends on the type of health indicator. A determination of unhealthiness is based on the criteria generates an alert.

Three types of health indicators are returned by the health monitor:

- **Threshold-based** indicators are measurements that represent a statistic (on a continuous range of values) of the behavior of the object. Warning and alarm threshold values define boundaries or zones for normal, warning, and alarm ranges. Threshold-based health indicators have three valid states: Normal, Warning, or Alarm.
- **State-based** indicators are measurements that represent a finite set of two or more distinct states of an object that defines whether the database object or resource is operating normally. One of the states is normal and all others are considered non-normal. State-based health indicators have two valid states: Normal, Attention.
- **Collection state-based** indicators are database-level measurements that represent an aggregate state or one or more objects within the database. Data is captured for each object in the collection and the highest severity of conditions among those objects is represented in the aggregated state. If one or more objects in the

collection are in a state requiring an alert, the health indicator shows Attention state. Collection state-based health indicators have two valid states: Normal, Attention.

Health indicators exist at the instance, database, table space, and table space container level.

You can access health monitor information through the Health Center, the CLP, or APIs. You can configure health indicators through these same tools.

An alert is generated in response to either a change from a normal to a non-normal state or a change in the health indicator value to a warning or alarm zone that is based on defined threshold boundaries. There are three types of alerts: attention, warning, and alarm.

- For health indicators measuring distinct states, an attention alert is issued if a non-normal state is registered.
- For health indicators measuring a continuous range of values, threshold values define boundaries or zones for normal, warning and alarm states. For example, if the value enters the threshold range of values that defines an alarm zone, an alarm alert is issued to indicate that the problem needs immediate attention.

The health monitor will only send notification and run an action on the first occurrence of a particular alert condition for a given health indicator. If the health indicator stays in a particular alert condition, no further notification will be sent and no further actions will be run. If the health indicator changes alert conditions, or goes back to normal state and re-enters the alert condition, notification will be sent anew and actions will be run.

The following table shows an example of a health indicator at different refresh intervals and the health monitor response to the health indicator state. This example uses the default warning of 80% and alarm thresholds of 90%.

*Table 45. Health indicator conditions at different refresh intervals*

| Refresh interval | Value of ts.ts_util (Table space utilization) health indicator | State of ts.ts_util health indicator | Health monitor response                                                        |
|------------------|----------------------------------------------------------------|--------------------------------------|--------------------------------------------------------------------------------|
| 1                | 80                                                             | warning                              | notification of warning is sent, actions for a warning alert condition are run |
| 2                | 81                                                             | warning                              | no notification is sent, no actions are run                                    |
| 3                | 75                                                             | normal                               | no notification is sent, no actions are run                                    |
| 4                | 85                                                             | warning                              | notification of warning is sent, actions for a warning alert condition are run |
| 5                | 90                                                             | alarm                                | notification of alarm is sent, actions for an alarm condition are run          |

**Health indicator process cycle:**

The following diagram illustrates the evaluation process for health indicators. The set of steps runs every time the refresh interval for the specific health indicator elapses.

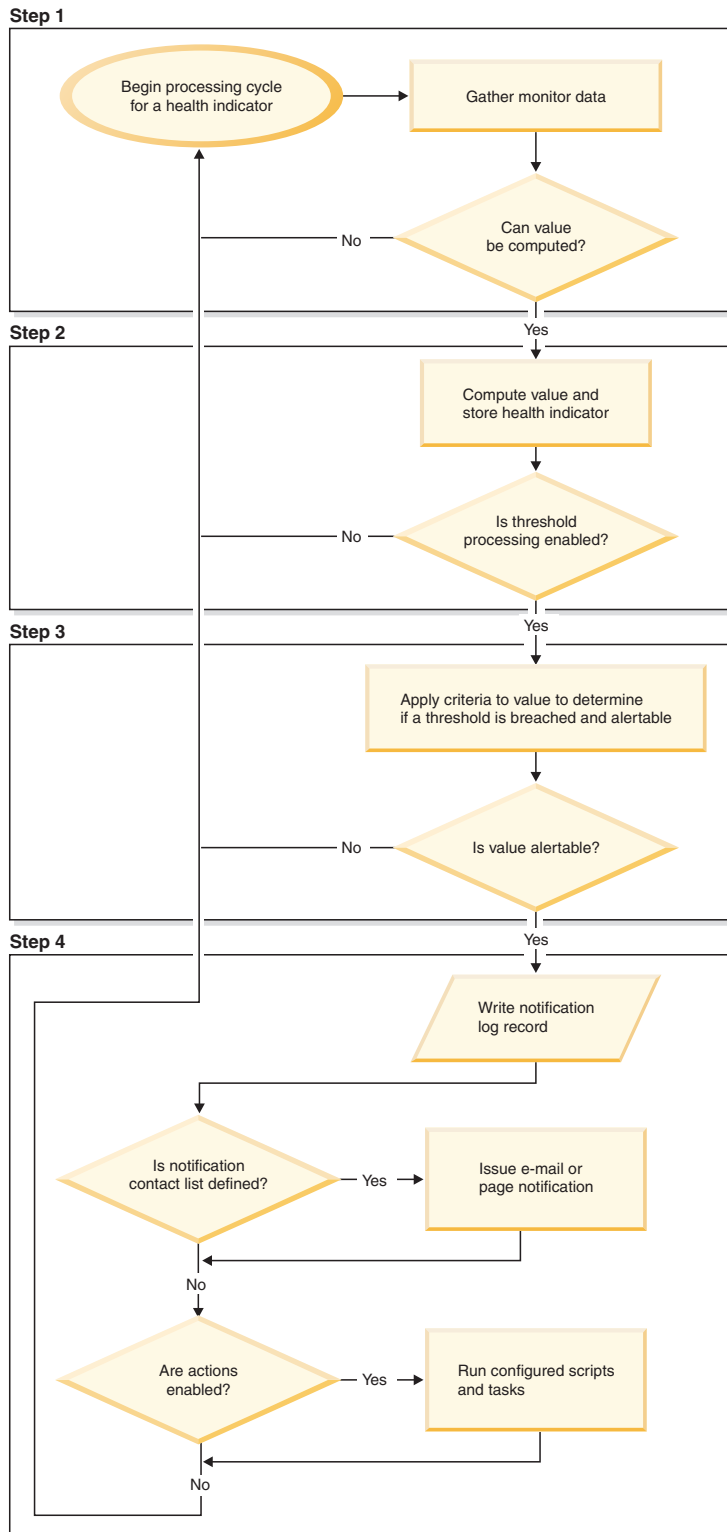


Figure 4. Health indicator process cycle



**Note:**

1. The NOTIFYLEVEL database manager configuration parameter controls whether alert notifications are sent to the DB2 administration notification log and to any defined contacts. A minimum severity level of 2 is required for alarm notifications. A minimum severity level of 3 is required for warnings and attention alerts to be sent.

**Enabling health alert notification:**

To enable e-mail or pager notification when an alert is generated, you must set configuration parameters and specify contact information.

The DB2 Administration Server (DAS) must be running on the system where the contact list is located. For example, if the CONTACT\_HOST configuration parameter is set to a remote system, the DAS must be running on the remote system in order for contacts to be notified of alerts.

To enable health alert notification:

1. Specify the SMTP\_SERVER parameter. The DAS configuration parameter, SMTP\_SERVER, specifies the location of the mail server to use when sending both e-mail and pager notification messages. Omit this step if the system where the DB2 database is installed is enabled as an unauthenticated SMTP server.
2. Specify the CONTACT\_HOST parameter. The DAS configuration parameter, CONTACT\_HOST, specifies the remote location of the contact list for all instances on the local system. By setting this parameter, a single contact list can be shared between multiple systems. Omit this step if you want to keep the contact list on the local system where the DB2 database is installed.
3. Specify the default contact for health monitor notification. To enable e-mail or pager notification from the health monitor when an alert is generated, a default administration contact must be specified. If you choose not to provide this information, notification messages cannot be sent for alert conditions. You can provide the default administration contact information during installation, or you can defer the task until after installation is complete. If you choose to defer the task or want to add more contacts or groups to the notification list, you can specify contacts through the CLP, C APIs, or the Health Center:

- 

**To specify contacts using the CLP:**

To define an e-mail contact as the default for health monitor notification, issue the following commands:

```
DB2 ADD CONTACT contact_name TYPE EMAIL ADDRESS  
      email_address DESCRIPTION 'Default Contact'
```

```
DB2 UPDATE NOTIFICATION LIST ADD CONTACT contact_name
```

For complete syntax details, see the Command Reference.

- 

**To specify contacts using C APIs:**

The following C code excerpt illustrates how to define health notification contacts:

```
...  
#include <db2ApiDf.h>  
  
SQL_API_RC rc = 0;  
struct db2AddContactData addContactData;  
struct sqlca sqlca;
```

```

char* userid = "myuser";
char* password = "pwd";
char* contact = "DBA1";
char* email = "dba1@mail.com";
char* desc = "Default contact";

memset(&addContactData, '\0', sizeof(addContactData));
memset (&sqlca, '\0', sizeof(struct sqlca));

addContactData.piUserid = userid;
addContactData.piPassword = password;
addContactData.piName = contact;
addContactData.iType = DB2CONTACT_EMAIL;
addContactData.piAddress = email;
addContactData.iMaxPageLength = 0;
addContactData.piDescription = desc;

rc = db2AddContact(db2Version810, &addContactData, &sqlca);

if (rc == 0) {
    db2HealthNotificationListUpdate update;
    db2UpdateHealthNotificationListData data;
    db2ContactTypeData contact;

    contact.pName = contact;
    contact.contactType = DB2CONTACT_EMAIL;

    update.iUpdateType = DB2HEALTHNOTIFICATIONLIST_ADD;
    update.piContact = &contact;

    data.iNumUpdates = 1;
    data.piUpdates = &update;

    rc = db2UpdateHealthNotificationList (db2Version810, &data, &ca);
}
...

```

#### To specify contacts using the Health Center:

- a. Right-click the instance for which you want to define the health notification list.
- b. Click **Configure**, then click **Alert Notification**. The Configure Health Alert Notification window opens.
- c. If contacts do not appear in the left side of the window in the **Available** list, click **Manage Contacts**. The Contacts window opens with the system name preselected.
- d. Click **Add Contact**. The Add Contact window opens.
- e. Define a contact by supplying a name and an e-mail address. Select **Address is for a pager** if the specified e-mail address is for a pager.
- f. Click **OK**.
- g. Close the Contacts window and return to the Configure Health Alert Notification window. The new contact now appears in the **Contacts available** list.
- h. Move the contact to the **Health notification contact list** by clicking the right arrow button.
- i. Click **OK** to include the contact in the health notification list.

#### Recommendation

If you are experiencing difficulties with notification, select

**Troubleshoot** below the Health notification contact list.  
The Troubleshoot Health Alert Notification wizard opens.

## Health Center overview

Use the Health Center to analyze and improve the health of DB2.

The following are examples of conditions that define what makes DB2 healthy:

- There are sufficient resources, such as free memory, table space containers, or logging storage, to accomplish tasks.
- Resources are used efficiently.
- Tasks complete within acceptable periods of time or without significant degradations in performance.
- Resources or database objects are not left indefinitely in unusable states.

**Important:** The Health Center has been deprecated in Version 9.7 and might be removed in a future release. For more information, see the “Control Center tools and DB2 administration server (DAS) have been deprecated” topic in the *What’s New for DB2 Version 9.7* book.

From the Health Center you can also open other centers and tools to help you investigate and maintain the health of your database.

To open the Health Center on Intel® platforms, from the **Start** menu, click **Start** → **Programs** → **IBM DB2** → **Monitoring Tools** → **Health Center**.

To open the Health Center using the command line on Intel platforms, run the following command:

```
db2hc
```

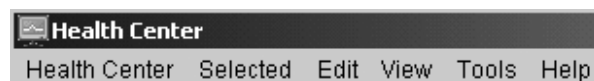
The following list categorizes some of the key tasks that you can do with the Health Center:

- “Enabling health alert notification” on page 130
  - Specifying contact settings and notification configuration parameters
  - Troubleshooting health alert notification
- “Configuring health indicators using the Health Center” on page 156
  - Enabling and disabling health indicator evaluation
  - Changing alert threshold and sensitivity settings
  - Running tasks and scripts when an alert occurs
- “Resolving health monitor alerts using the Health Center” on page 150
  - Using the Recommendation advisor to select and implement recommendations

## The Health Center interface

The Health Center interface has the following elements that help you determine and resolve problems related to the overall health of your system.

### Health Center Menu bar



Use the menu bar to work with objects in the Health Center, open other administration centers and tools, and access online help.

The Health Center menu bar contains the following menus:

### Health Center toolbar



Use the toolbar icons below the menu bar to access other centers and tools, and to refresh the content view of the Health Center.

### Toggle buttons



Use the toggle buttons to select the alert states that appear in the Navigation view. Each button corresponds to a minimum alert severity that a database object needs in order to appear in the view. Selecting a different button only affects the display and does not affect the object itself.



Shows objects in alarm state



Shows objects in alarm and warning states



Shows objects in any alert state: alarm, warning, attention, normal, and not monitored.



Shows all objects

### Navigation view



Use the navigation view to display and work with instance and database objects. When you select an object in the Navigation view, the current alerts for that object and all its children are displayed in the Alerts view. To change the level of alert that object must have before the navigation view displays it, right click in the navigation view away from the listed objects. This will open a pop-up menu of alert levels. Select the alert levels that you want displayed. You can also choose what alert levels to display by clicking the toggle buttons.

### Alerts view

Use the Alerts view to display and work with current alerts. The Alerts view displays those alerts that currently exist for the object and its children database objects selected in the navigation view. For example, if you select an instance, alerts display for the instance and all its databases and table spaces. If you select a database, alerts display for the database and all table spaces for the database. Select and right-click one or more alerts in the Alerts view to invoke actions for them.

## Alerts view toolbar



Use the toolbar below the Alerts view to tailor the view of alerts in the Alerts view to suit your needs.

## Investigating alert conditions:

### Health monitor

The health monitor captures information about the database manager, database, table space and table space containers. The health monitor calculates health indicators based on data retrieved from database system monitor elements, the operating system, and DB2 database. The health monitor can only evaluate health indicators on a database and its objects when the database is active. You can keep the database active either by starting it with the `ACTIVATE DATABASE` command or by maintaining a permanent connection to the database.

The health monitor retains a maximum of ten history records for each health indicator. This history is stored in the `<instance path>\hmonCache` directory and is removed when the health monitor is stopped. The health monitor automatically prunes obsolete history records when the maximum number of records has been reached.

Health monitor data is accessible through health snapshots. Each health snapshot reports the status for each health indicator based on its most recent refresh interval. The snapshots are useful for detecting existing database health problems and predicting potential poor health of the database environment. You can capture a health snapshot from the CLP, by using APIs in a C or C++ application, or by using the graphical administration tools.

Health monitoring requires an instance attachment. If an attachment to an instance has not been established using the `ATTACH TO` command, then a default instance attachment to the local instance is created.

In partitioned database environments, snapshots can be taken at any partition of the instance, or globally using a single instance connection. A global snapshot aggregates the data collected at each partition and returns a single set of values.

### Usage notes

The health monitor is supported on all editions of the DB2 database.

To start or stop the health monitor from the Health Center, right-click an instance in the Health Center navigational view and select Start Health Monitor or Stop Health Monitor

On Windows, the service for the DB2 instance needs to run under an account with SYSADM authority. You can use the `-u` option on the `db2icrt` command, or use the Services folder on Windows and edit the Log On properties to use an account with administrator privilege.

The health monitor process runs as a DB2 fenced mode process. These processes appear as DB2FMP on Windows. On other platforms, the health monitor process appears as DB2ACD.

The DB2 Administration server must be running on the system where the health monitor resides for notifications to be sent and alert actions to be run. If remote scripts, tasks, or contact lists are used, the DB2 Administration server on the remote system must also be started.

The tools catalog database is required only for creating tasks. If you do not use alert task actions for any health indicator, the tools catalog database is not required by the health monitor.

If you fall back to DB2 UDB Version 8.1 from a later version of the DB2 database system, any registry changes that have been made are lost. The registry reverts to the version 8.1 HealthRules.reg file that contains the settings that existed before you upgraded and started using the settings in the newer registry file.

**Health indicator data:** The health monitor records a set of data for each health indicator on each database partition, including:

- Health indicator name
- Value
- Evaluation timestamp
- Alert state
- Formula, if applicable
- Additional information, if applicable
- History of up to ten of the most recent health indicator evaluations. Each history entry captures the following health indicator evaluations leading up to the current health indicator output:
  - Value
  - Formula (if applicable)
  - Alert state
  - Timestamp

The health monitor also tracks the highest severity alert state at the instance, database, and table space levels. At each level, this health indicator represents the highest severity alert existing for health indicators at that level, or any of the levels below it. For example, the highest severity alert state for an instance includes health indicators on the instance, any of its database, and any of the table spaces and table space containers for each of the databases.

### **Capturing database health snapshots:**

*Capturing a database health snapshot using SQL table functions:*

You can capture database health snapshots using SQL table functions. Each available health snapshot table function corresponds to a health snapshot request type.

To capture a database health snapshots using SQL table functions:

1. Identify the SQL table function you plan to use.  
SQL table functions have two input parameters:
  - A VARCHAR(255) for the database name
  - An INT for the partition number (a value between 0 and 999). Enter the integer corresponding to the partition number you want to monitor. To

capture a snapshot for the currently connected partition, enter a value of -1.  
To capture a global snapshot, enter a value of -2.

**Note:** The database manager snapshot SQL table functions are the only exception to this rule because they have only one parameter. The single parameter is for partition number. If you enter NULL for the database name parameter, the monitor uses the database defined by the connection through which the table function has been called.

2. Issue the SQL statement.

The following example captures a basic health snapshot for the currently connected partition, and on the database defined by the connection from which this table function call is made:

```
SELECT * FROM TABLE( HEALTH_DB_INFO( cast (NULL as VARCHAR(1)), -1))  
as HEALTH_DB_INFO
```

You can also select individual monitor elements from the returned table. Each column in the returned table corresponds to a monitor element. Accordingly, the monitor element column names correspond directly to the monitor element names. The following statement returns only the db path and server platform monitor elements:

```
SELECT db_path, server_platform  
FROM TABLE( HEALTH_DB_INFO( cast (NULL as VARCHAR(1)), -1 ) )  
as HEALTH_DB_INFO
```

*Capturing a database health snapshot using the CLP:*

You can capture health snapshots using the GET HEALTH SNAPSHOT command from the CLP. The command syntax supports retrieval of health snapshot information for the different object types monitored by the health monitor.

You must have an instance attachment to capture a health snapshot. If there is not an attachment to an instance, then a default instance attachment is created. To obtain a snapshot of a remote instance, you must first attach to that instance.

To capture a database health snapshot using the CLP

1. From the CLP, issue the GET HEALTH SNAPSHOT command with the desired parameters.

In the following example, a database manager level health snapshot is captured immediately after starting the database manager.

```
db2 get health snapshot for dbm
```

2. For partitioned database systems, you can capture a database snapshot specifically for a certain partition or globally for all partitions. To capture a health snapshot for a database on a specific partition (for example, partition number 2), issue the following command:

```
db2 get health snapshot for db on sample at dbpartitionnum 2
```

To capture a database snapshot for all applications on all partitions, issue the following command:

```
db2 get health snapshot for db on sample global
```

The following command captures a health snapshot with additional detail, including the formula, additional information, and health indicator history:

```
db2 get health snapshot for db on sample show detail
```

3. For collection state-based health indicators, you can capture a database snapshot for all collection objects, regardless of state. The regular GET

HEALTH SNAPSHOT FOR DB command returns all collection objects requiring an alert for all collection state-based health indicators.

To capture a health snapshot for a database with all collection objects listed, issue the following command:

```
db2 get health snapshot for db on sample with full collection
```

*Capturing a database health snapshot from a client application:*

You can capture health snapshots using the snapshot monitor API in a C or C++ application. A number of different health snapshot request types can be accessed by specifying parameters in the db2GetSnapshot API.

You must be attached to an instance to capture a health snapshot. If there is not an attachment to an instance, then a default instance attachment is created. To obtain a snapshot of a remote instance, you must first attach to that instance.

1. Include the sqlmon.h and db2ApiDf.h DB2 libraries in your code. These libraries are found in the sqllib\include directory.

```
#include <db2ApiDf.h>
#include <sqlmon.h>
```

2. Set the snapshot buffer unit size to 50 KB.

```
#define SNAPSHOT_BUFFER_UNIT_SZ 51200
```

3. Declare the sqlma, sqlca, sqlm\_collected, and db2GetSnapshotData structures.

```
struct sqlma *pRequestedDataGroups;
struct sqlca sqlca;
memset (&sqlca, '\0', sizeof(struct sqlca));
struct sqlm_collected collectedData;
memset (&sqlm_collected, '\0', sizeof(struct sqlm_collected));
db2GetSnapshotData getSnapshotParam;
memset(&db2GetSnapshotData, '\0', sizeof(db2GetSnapshotData));
```

4. Initialize a pointer to contain the snapshot buffer, and to establish the buffer's size.

```
static sqluint32 snapshotBufferSize = SNAPSHOT_BUFFER_UNIT_SZ;
sqluint32 outputFormat;
char *snapshotBuffer;
```

5. Initialize the sqlma structure and specify that the snapshot you are capturing is of database manager level information.

```
pRequestedDataGroups = (struct sqlma *)malloc(SQLMASIZE(1));
memset(&pRequestedDataGroups, '\0', sizeof(struct pRequestedDataGroups));
pRequestedDataGroups->obj_num = 1;
pRequestedDataGroups->obj_var[0].obj_type = SQLMA_DB2;
```

6. Initialize the buffer, which will hold the snapshot output.

```
snapshotBuffer = (char *)malloc(snapshotBufferSize);
memset (&snapshotBuffer, '\0', sizeof(snapshotBuffer));
```

7. Populate the db2GetSnapshotData structure with the snapshot request type (from the sqlma structure), buffer information, and other information required to capture a snapshot.

```
getSnapshotParam.piSqlmaData = pRequestedDataGroups;
getSnapshotParam.poCollectedData = &collectedData;
getSnapshotParam.poBuffer = snapshotBuffer;
getSnapshotParam.iVersion = SQLM_DBMON_VERSION9_5;
getSnapshotParam.iBufferSize = snapshotBufferSize;
getSnapshotParam.iStoreResult = 0;
getSnapshotParam.iNodeNumber = SQLM_CURRENT_NODE;
getSnapshotParam.poOutputFormat = &outputFormat;
getSnapshotParam.iSnapshotClass = SQLM_CLASS_HEALTH;
```



8. Capture the health snapshot. Pass the following parameters:
  - db2GetSnapshotData structure, which contains the information necessary to capture a snapshot
  - A reference to the buffer where snapshot output is directed.

```
db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
```

9. Include logic to handle buffer overflow. After a snapshot is taken, the sqlcode is checked for a buffer overflow. If a buffer overflow occurs, the buffer is cleared, reinitialized, and the snapshot is taken again.

```
while (sqlca.sqlcode == 1606)
{
    free(snapshotBuffer);
    snapshotBufferSize += SNAPSHOT_BUFFER_UNIT_SZ;
    snapshotBuffer = (char *)malloc(snapshotBufferSize);
    if (snapshotBuffer == NULL)
    {
        printf("\nMemory allocation error.\n");
        return;
    }

    getSnapshotParam.iBufferSize = snapshotBufferSize;
    getSnapshotParam.poBuffer = snapshotBuffer;
    db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
}
```

10. Process the snapshot monitor data stream. Refer to the figure following these steps to see the snapshot monitor data stream.

11. Clear the buffer.

```
free(snapshotBuffer);
free(pRequestedDataGroups);
```

After you capture a health snapshot with the db2GetSnapshot API, the API returns the health snapshot output as a self-describing data stream. The following example shows the data stream structure:

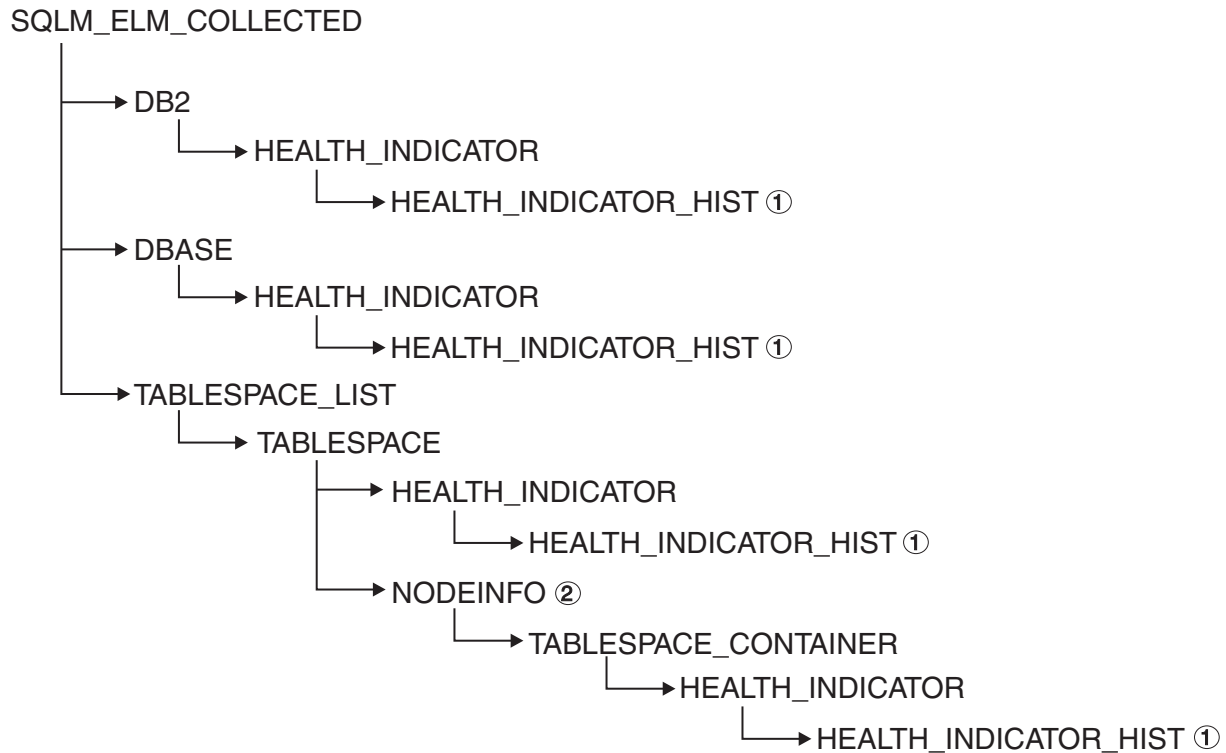


Figure 5. Health snapshot self-describing data stream

**Legend:**

1. Only available when the SQLM\_CLASS\_HEALTH\_WITH\_DETAIL snapshot class is used.
2. Only available in DB2 Enterprise Server Edition. Otherwise, table space container stream follows.

The following hierarchies display the specific elements in the health snapshot self-describing data stream.

The hierarchy of elements under SQLM\_ELM\_HI:

```

SQLM_ELM_HI
  SQLM_ELM_HI_ID
  SQLM_ELM_HI_VALUE
  SQLM_ELM_HI_TIMESTAMP
  SQLM_ELM_SECONDS
  SQLM_ELM_MICROSEC
  SQLM_ELM_HI_ALERT_STATE
  
```

The hierarchy of elements under SQLM\_ELM\_HI\_HIST, available only with the SQLM\_CLASS\_HEALTH\_WITH\_DETAIL snapshot class:

```

SQLM_ELM_HI_HIST
  SQLM_ELM_HI_FORMULA
  SQLM_ELM_HI_ADDITIONAL_INFO
  SQLM_ELM_HEALTH_INDICATOR_HIST
  SQLM_ELM_HI_ID
  SQLM_ELM_HI_VALUE
  SQLM_ELM_HI_TIMESTAMP
  SQLM_ELM_SECONDS
  
```

```

        SQLM_ELM_MICROSEC
    SQLM_ELM_HI_ALERT_STATE
    SQLM_ELM_HI_FORMULA
    SQLM_ELM_HI_ADDITIONAL_INFO

```

The hierarchy of elements under SQLM\_ELM\_OBJ\_LIST:

```

SQLM_ELM_HI_OBJ_LIST
  SQLM_ELM_HI_OBJ_NAME
  SQLM_ELM_HI_OBJ_DETAIL
  SQLM_ELM_HI_OBJ_STATE
  SQLM_ELM_HI_TIMESTAMP
    SQLM_ELM_SECONDS
    SQLM_ELM_MICROSEC

```

The hierarchy of elements under SQLM\_ELM\_OBJ\_LIST\_HIST, available only with the SQLM\_CLASS\_HEALTH\_WITH\_DETAIL snapshot class:

```

SQLM_ELM_HI_OBJ_LIST_HIST
  SQLM_ELM_HI_OBJ_NAME
  SQLM_ELM_HI_OBJ_STATE
  SQLM_ELM_HI_TIMESTAMP
    SQLM_ELM_SECONDS
    SQLM_ELM_MICROSEC

```

### Health monitor sample output:

The following examples show health snapshots taken using the CLP, and their corresponding output, and illustrate the nature of the health monitor. The objective in the examples is to check the overall health status immediately after starting the database manager.

1. Take the database manager snapshot, using the GET HEALTH SNAPSHOT command:

```
db2 get health snapshot for dbm
```

After the GET HEALTH SNAPSHOT command is issued from the CLP, the snapshot output is directed to the screen.

```

Node name                =
Node type                 = Database Server with local
                        and remote clients
Instance name             = DB2
Snapshot timestamp       = 11-07-2002 12:43:23.613425

Number of database partitions in DB2 instance = 1
Start Database Manager timestamp = 11-07-2002 12:43:18.000108
Instance highest severity alert state = Not yet evaluated

```

Health Indicators:

```
    Not yet evaluated
```

2. Analyze the output. From this health snapshot, you can see that the instance highest severity alert state is "Not yet evaluated". The instance is in this state because the health monitor has just started and has not yet evaluated any health indicators.

Should the instance highest severity alert state not change:

- Check the value of the HEALTH\_MON database manager configuration parameter to determine if the health monitor is on.
- If HEALTH\_MON=OFF, then the health monitor is not started. To start the health monitor, issue the UPDATE DBM CFG USING HEALTH\_MON ON command.

- If HEALTH\_MON=ON, attach to the instance to activate the health monitor. If an instance attachment exists, it is possible that the health monitor could not be loaded into memory.

Another example of taking a database health snapshot using the CLP is outlined below.

1. Before you begin, ensure that a database connection exists, and that the database is quiesced.
2. Take the database manager snapshot, using the GET HEALTH SNAPSHOT command:

```
db2 get health snapshot for db on sample
```

3. After the GET HEALTH SNAPSHOT command is issued from the CLP, the snapshot output is directed to the screen.

```

Database Health Snapshot

Snapshot timestamp                = 12-09-2002 11:44:37.793184

Database name                     = SAMPLE
Database path                    = E:\DB2\NODE0000\SQL00002\
Input database alias              = SAMPLE
Operating system running at database server= NT
Location of the database          = Local
Database highest severity alert state = Attention

```

Health Indicators:

```

...
Indicator Name                   = db.log_util
Value                           = 60
Unit                             = %
Evaluation timestamp             = 12-09-2002 11:44:00.095000
Alert state                      = Normal

Indicator Name                   = db.db_op_status
Value                           = 2
Evaluation timestamp             = 12-09-2002 11:44:00.095000
Alert state                      = Attention

```

4. Analyze the output.

This health snapshot reveals that there is an attention alert on the *db.db\_op\_status* health indicator. The value of 2 indicates that the database is in quiesced state.

### Global health snapshots:

On a partitioned database system you can take a health snapshot of the current partition, a specified partition, or all partitions. When taking a global health snapshot across all the partitions of a partitioned database, data is aggregated, where possible, before the results are returned.

The aggregated alert state for the health indicator is equivalent to the highest severity alert state across all the database partitions. Additional information and history data cannot be aggregated across the database partitions, and therefore are not available. The remaining data for the health indicator is aggregated as detailed in the table below.

Table 46. Aggregation of health indicator value, timestamp, and formula data

| Health indicator                                                                                                                                                                                                                                                                                                                                       | Aggregation details                                                                                                                                                        |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <ul style="list-style-type: none"> <li>• db2.db2_op_status</li> <li>• db2.sort_privmem_util</li> </ul>                                                                                                                                                                                                                                                 | The health indicator value is obtained from the partition that contains the highest value.                                                                                 |
| <ul style="list-style-type: none"> <li>• db2.mon_heap_util</li> <li>• db.db_op_status</li> <li>• db.sort_shrmem_util</li> <li>• db.spilled_sorts</li> <li>• db.log_util</li> <li>• db.log_fs_util</li> <li>• db.locklist_util</li> <li>• db.apps_waiting_locks</li> <li>• db.db_heap_util</li> <li>• db.db_backup_req</li> <li>• ts.ts_util</li> </ul> | The evaluation timestamp and formula are obtained from the same partition.                                                                                                 |
| <ul style="list-style-type: none"> <li>• db.max_sort_shrmem_util</li> <li>• db.pkgcache_hitratio</li> <li>• db.catcache_hitratio</li> <li>• db.shrworkspace_hitratio</li> </ul>                                                                                                                                                                        | The health indicator value is obtained from the partition that contains the lowest value.                                                                                  |
| <ul style="list-style-type: none"> <li>• db.deadlock_rate</li> <li>• db.lock_escal_rate</li> </ul>                                                                                                                                                                                                                                                     | The health indicator value is the sum of the values across all the database partitions.                                                                                    |
| <ul style="list-style-type: none"> <li>• ts.ts_op_status</li> <li>• tsc.tscont_op_status</li> <li>• tsc.tscont_util</li> </ul>                                                                                                                                                                                                                         | The evaluation timestamp and formula cannot be aggregated and are not available.                                                                                           |
| <ul style="list-style-type: none"> <li>• ts.ts_op_status</li> <li>• tsc.tscont_op_status</li> <li>• tsc.tscont_util</li> </ul>                                                                                                                                                                                                                         | These health indicators is not aggregated.                                                                                                                                 |
| <ul style="list-style-type: none"> <li>• db.hadr_op_status</li> <li>• db.hadr_log_delay</li> </ul>                                                                                                                                                                                                                                                     | These health indicators are not supported in a multiple partition database.                                                                                                |
| <ul style="list-style-type: none"> <li>• db.tb_reorg_req</li> <li>• db.tb_runstats_req</li> <li>• db.fed_nicknames_op_status</li> <li>• db.fed_servers_op_status</li> </ul>                                                                                                                                                                            | This health indicator is evaluated only on one partition, so no aggregation is required. The data is returned from the partition which is evaluating the health indicator. |

**Note:** When taking a global snapshot on a single partition object, the output includes all the attributes because there are no partitions to aggregate.

### Graphical tools for the health monitor: Health Center

The Health Center is a graphical administration tool designed to support management-by-exception. For all Windows, Linux, and UNIX instances and databases cataloged on the client, the Health Center provides:

- A central location to view the rolled up alert state of all instances and their databases
- A graphical interface to view current alerts on the instances and databases and their children objects

- A graphical interface to access details and recommended resolution actions for current alerts

To start the Health Center from the command line, type the db2hc command.

**Important:** The Health Center has been deprecated in Version 9.7 and might be removed in a future release. For more information, see the “Control Center tools and DB2 administration server (DAS) have been deprecated” topic in the *What’s New for DB2 Version 9.7* book.

On Windows, you can also start the Health Center from the Start Menu by clicking **Start** → **Programs** → **IBM DB2** → <DB2 copy name> → **Monitoring Tools** → **Health Center**.

The Health Center has a navigation tree in the left panel and an Alerts view in the right panel. The contents of the navigation view are filtered based on the toggle button selected at the top of the navigation view.

The Health Center opens with the **Object in Any Alert State** toggle button selected, which helps to identify those instances with current alerts that should be addressed. When the **All Objects** toggle button is selected, all Windows, Linux, and UNIX instances cataloged on the client and their respective states are displayed. Instances without an icon do not have the health monitor running or are instances prior to version 8, which lack support for health monitor functionality.

When you select an instance, the Health Center requests status from the health monitor for the selected instance. The Alerts view fills with all current alerts for the instance, any of its databases, and any of the table spaces and table space containers of each database. If you expand the instance in the navigation view and select a child database object, the Alerts view is restricted to alerts for the selected database and any of its table spaces or table space containers.

The refresh icon is located in the upper-right corner of the Health Center. Clicking the refresh icon for immediate refresh, or setting a particular refresh interval, causes the Health Center to query the health monitor on the server for its current status. This query does not cause the health monitor to refresh the health indicator evaluations. Each health indicator has a defined refresh interval. Only when the refresh interval has passed will the health indicator be reevaluated for alert state. Only the current status of the health indicators is shown on each timed refresh or requested refresh of the Health Center.

The Alerts view has a function to define customized views with specific customized columns and sorting orders. There are six predefined views in the Health Center that you can customize to your personal naming and categorization scheme. You can select the predefined views by using the toolbar at the bottom of the window or by selecting **Saved Views** in the **View** menu. To define your own customized views, click the **View** button on the toolbar at the bottom of the window, or use the **View** menu. The view that is selected for displaying data in the Alerts view is remembered on the next invocation of the Health Center.

To get the details for an alert, select the alert row in the Alerts view. Using the **Selected** menu, or by right-clicking the row, select **Show Details**. The Details window shows the detailed information for the alert including the object and partition where the alert occurred, the formula (if applicable), and value for the health indicator.

For threshold-based health indicators, the thresholds that were used in determining the alert condition are displayed. The Details window also displays additional information for the health indicator. This information might include values for configuration parameters or other monitor data that provides context for the alert. A description of the health indicator is displayed, including the purpose for the health indicator and why it is an important attribute to measure.

For collection state-based health indicators, the list of collection objects is displayed in the Objects in **Health Indicator Alert State** table. Object name, state, timestamp, and details are provided in the table.

A View History button is provided on the details page. History records are stored for the health indicator starting with the second refresh of the health indicator evaluation. Content is displayed in the View History dialog in the Health Center only after the history records are stored. The history of collection objects, for collection state-based health indicators, can be viewed by clicking the **View Collection History** button in the History window.

### **Health Center Status Beacon**

The Health Center Status Beacon is a visual indicator that can be enabled in the DB2 administration tools. When the Health Center is not open, the beacon will notify you of current alerts while you are working with other DB2 administration tools. The beacon is intended to prompt the user to open the Health Center because of an alert condition.

The Health Center Status Beacon has two different notification methods. One notification method uses a pop-up message. Another notification method uses a graphical beacon that displays on the right portion of the status line of open windows. The graphical beacon includes a button that provides single-click access to the Health Center.

Both beacon notification methods are enabled through the Tools Settings dialog. The "notify through pop-up" method controls the pop-up message notification, and the "notify through status line" method controls the visual beacon.

### **Retrieving health recommendations:**

*Health recommendation queries with SQL:*

Recommendations can be queried with SQL using the SYSPROC.HEALTH\_HI\_REC stored procedure.

When using the SYSPROC.HEALTH\_HI\_REC stored procedure, recommendations are returned in an XML document that is:

- Formatted according to the health recommendations XML schema DB2RecommendationSchema.xsd located in the sqllib\misc directory.
- Encoded in UTF-8 and contains text in the client language.
- Organized as a collection of recommendation sets, where each recommendation set describes a problem (health indicator) being resolved and contains one or more recommendations to resolve that health indicator. Refer to the schema definition for specific details about information that can be retrieved from the document.

All information available through the CLP is also available in the XML recommendation document that is returned when you query with SQL.

The SYSPROC.HEALTH\_HI\_REC stored procedure takes the following arguments:

- A health indicator
- A definition of the object on which the health indicator has entered an alert state

The output recommendation document is returned as a BLOB. Therefore, it is not helpful to work with this stored procedure from the command line, since the CLP will limit the amount of output displayed. It is recommended that this stored procedure be invoked using a high level language (such as C or Java) that allows the returned XML document to be properly parsed to retrieve any required elements and attributes.

*Retrieving health recommendations using the CLP:*

Recommendations can be retrieved using the GET RECOMMENDATIONS command from the CLP. The command syntax supports querying recommendations to resolve a specific health alert, such as a health indicator that has currently entered an alert state on a particular object.

You must have an instance attachment to retrieve recommendations from the health monitor. If there is not an attachment to an instance, a default instance attachment is created. To obtain recommendations from a health monitor on a remote instance, you must first attach to that instance. No special authority is required to retrieve recommendations from the health monitor.

The command syntax also supports retrieval of the complete set of recommendations for a given health indicator, which does not have to be in an alert state when the command is executed. Recommendations for resolving an alert on a specific health indicator can be queried at either a single partition level or a global level.

When querying recommendations on a health alert on a specific object, the health monitor is solving a specific alert and is able to provide details on the alert being resolved in the problem section of the output.

The health monitor is also able to provide a ranking for the recommendations and, in some cases, it might be able to generate scripts that can be executed to resolve the alert. Additionally, the health monitor might reject and not display some recommendations if they are not applicable to the particular problem situation. On the other hand, if recommendations are queried by health indicator name only, as in the first example below, the total set of possible recommendations will always be returned. In such cases, the CLP command is simply providing information about actions that a user should consider undertaking if they see an alert.

Retrieve the recommendations using the GET RECOMMENDATIONS command:

1. You might want to issue the following command to see the total set of actions that could be recommended to resolve an alert on the **db.db\_op\_status** health indicator.

```
db2 get recommendations for health indicator db.db_op_status
```

In this example, the full set of recommendations for the **db.db\_op\_status** health indicator is returned. The health indicator does not have to be in an alert state to issue this command.

This output shows that there are two possible recommendations for this health indicator: unquiesce the database or investigate rollforward progress on the database. Because the command is being used to query all possible



recommendations, rather than to ask how to resolve a specific alert, the health monitor cannot identify the best recommendation in this case. As a result, the full set of recommendations is returned.

Recommendations:

Recommendation: Investigate rollforward progress.

A rollforward is in progress on the database due to an explicit request from the administrator. You have to wait for the rollforward to complete for the instance to return to active state.

Take one of the following actions:

Launch DB2 tool: Utility Status Manager

The Utility Status Manager allows you to monitor the progress and change the priority of currently running utilities.

To open the Utility Status Manager:

1. From the Control Center, expand the object tree until you find the database that you want.
2. Right-click the database, and click Manage Utilities from the pop-up menu. The Utility Status Manager opens.

To view progress of the rollforward utility, right-click on the rollforward utility and select View Progress Details.

From the Command Line Processor, issue the commands shown in the following example to view the progress of the rollforward utility:

```
LIST UTILITIES SHOW DETAIL
```

Recommendation: Unquiesce the database.

The database has been put into QUIESCE PENDING or QUIESCE state by an explicit request from the administrator. If you have QUIESCE\_CONNECT authority, or are DBADM or SYSADM, you will still have access to the database and will be able to use it normally. For all other users, new connections to the database are not permitted and new units of work cannot be started. Also, depending on the quiesce request, active units of work will be allowed to complete or will be rolled back immediately. You can issue an unquiesce to return to active state.

Take one of the following actions:

Launch DB2 tool: Control Center Unquiesce Database

The Control Center has an option on a database that can be used to unquiesce the database.

To unquiesce a database:

1. From the Control Center, expand the object tree until you find the database that you want.
2. Right-click the database, and click Unquiesce from the pop-up menu. The database is unquiesced.

From the Command Line Processor, issue the commands shown in the following example:

```
CONNECT TO DATABASE database-alias  
UNQUIESCE DATABASE
```

2. Suppose you observe that the health indicator **db.db\_heap\_util** has entered an alert state for the database SAMPLE, and you want to determine how to resolve the alert. In this case, you want to resolve a specific problem, therefore you could issue the GET RECOMMENDATIONS command in the following way:

```
db2 get recommendations for health indicator db.db_heap_util
    for database on sample
```

This output shows a summary of the problem and a set of recommendations to resolve the problem. The health monitor has ranked the recommendations in its order of preference. Each recommendation contains a description and a set of actions that indicate how to perform the recommended action.

Problem:

|                        |                       |
|------------------------|-----------------------|
| Indicator Name         | = db.db_heap_util     |
| Value                  | = 42                  |
| Evaluation timestamp   | = 11/25/2003 19:04:54 |
| Alert state            | = Alarm               |
| Additional information | =                     |

Recommendations:

Recommendation: Increase the database heap size.

Rank: 1

Increase the database configuration parameter dbheap sufficiently to move utilization to normal operating levels. To increase the value, set the new value of dbheap to be equal to  $(\text{pool\_cur\_size} / (4096 * U))$  where U is the desired utilization rate. For example, if your desired utilization rate is 60% of the warning threshold level, which you have set at 75%, then  $U = 0.6 * 0.75 = 0.45$  (or 45%).

Take one of the following actions:

Execute the following scripts at the DB2 server (this can be done using the EXEC\_DB2\_CMD stored procedure):

```
CONNECT TO DATABASE SAMPLE;
UPDATE DB CFG USING DBHEAP 149333;
CONNECT_RESET;
```

Launch DB2 tool: Database Configuration Window

The Database Configuration window can be used to view and update database configuration parameters.

To open the Database Configuration window:

1. From the Control Center, expand the object tree until you find the databases folder.
2. Click the databases folder. Any existing database are displayed in the contents pane on the right side of the window.
3. Right-click the database that you want in the contents pane, and click Configure Parameters in the pop-up menu. The Database Configuration window opens.

On the Performance tab, update the database heap size parameter as suggested and click OK to apply the update.

Recommendation: Investigate memory usage of database heap.

Rank: 2

There is one database heap per database and the database manager uses it on behalf of all applications connected to the database. The data area is expanded as needed up to the maximum specified by dbheap.

For more information on the database heap, refer to the DB2 Information Center.

Investigate the amount of memory that was used for the database heap over time to determine the most appropriate value for the database heap configuration parameter. The database system monitor tracks the highest amount of memory that was used for the database heap.

Take one of the following actions:

Launch DB2 tool: Memory Visualizer

The Memory Visualizer is used to monitor memory allocation within a DB2 instance. It can be used to monitor overall memory usage, and to update configuration parameters for individual memory components.

To open the Memory Visualizer:

1. From the Control Center, expand the object tree until you find the instances folder.
2. Click the instances folder. Any existing instances are displayed in the contents pane on the right side of the window.
3. Right-click the instance that you want in the contents pane, and click View Memory Usage in the pop-up menu. The Memory Visualizer opens.

To start the Memory Visualizer from the command line issue the db2memvis command.

The Memory Visualizer displays a hierarchical list of memory pools for the database manager. Database Heap is listed under the Database Manager Memory group for each database. On Windows, it is listed under the Database Manager Shared Memory group.

Click the check box on the Show Plot column for the Database Heap row to add the element to the plot.

3. For partitioned database systems you can query recommendations for a health indicator that has entered an alert state on a certain partition, or globally for all partitions. When recommendations are queried globally, a set of recommendations is returned that applies to the health indicator on all partitions. For example, if the health indicator is in an alert state on partitions 1 and 3, a collection of two scripts might be returned where each script is to be applied to a different partition.

The following example shows how to query recommendations for a health indicator on a specific partition (in this example, partition number 2):

```
db2 get recommendations for health indicator db.db_heap_util
    for database on sample at dbpartitionnum 2
```

The following example shows how to retrieve a set of recommendations to resolve a health indicator that is in an alert state on several partitions:

```
db2 get recommendations for health indicator db.db_heap_util
    for database on sample global
```

*Retrieving health recommendations using a client application:*

Recommendations can be queried using the db2GetRecommendations API in a C or C++ application.

You must have an instance attachment to capture a health snapshot. If there is not an attachment to an instance, then a default instance attachment is created. To query recommendations on a remote instance, you must first attach to that instance.

When using the `db2GetRecommendations` API, recommendations are returned in an XML document that is:

- Formatted according to the health recommendations XML schema `DB2RecommendationSchema.xsd` located in the `MISC` subdirectory within the `SQLLIB` directory.
- Encoded in UTF-8 and contains text in the client language.
- Organized as a collection of recommendation sets, where each recommendation set describes a problem (health indicator) being resolved and contains one or more recommendations to resolve that health indicator. Refer to the schema definition for specific details about what information that can be retrieved from the document.

All information available through the CLP is also available in the XML recommendation document that is returned.

To retrieve health recommendations using a client application:

1. Include the `sqlmon.h` and `db2ApiDf.h` DB2 header files. These are found in the `sqllib\include` directory.

```
#include <db2ApiDf.h>
#include <sqlmon.h>
```

2. Declare the `sqlca`, and the `db2GetRecommendationsData` structure.

```
struct sqlca sqlca ;
db2GetRecommendationsData recData ;

memset( &sqlca, '\0', sizeof( struct sqlca ) ) ;
memset( &recData, '\0', sizeof( db2GetRecommendationsData ) ) ;
```

3. Populate the `db2GetRecommendationsData` structure with information about the alert for which you want to retrieve recommendations. In the code excerpt that follows, recommendations are being queried for the **db2.db\_heap\_util** health indicator on the Sample database.

```
recData.iSchemaVersion = DB2HEALTH_RECSCHEMA_VERSION8_2 ;
recData.iNodeNumber = SQLM_CURRENT_NODE ;
recData.iIndicatorID = SQLM_HI_DATABASE_HEAP_UTILIZATION ;
recData.iObjType = DB2HEALTH_OBJTYPE_DATABASE ;
recData.piDbName = "SAMPLE" ;
```

4. Invoke the `db2GetRecommendations` API to retrieve recommendations for an alert on this health indicator on the specified database.

```
db2GetRecommendations( db2Version820, &recData, &sqlca ) ;
```

5. Check the `sqlcode` returned in the `sqlca` for any errors that occurred. If the API call was successful, process the recommendation XML document that is returned in the `poRecommendation` field of the `db2GetRecommendationsData` structure. Use your choice of XML parser to extract the required elements or attributes. Refer to the `DB2RecommendationSchema.xsd` XML schema in the `sqllib\misc` directory for details about the information that can be retrieved from the XML document.

6. Free any memory allocated by the `db2GetRecommendations` API. This will free the recommendation document returned in the `poRecommendation` field of the `db2GetRecommendationsData` structure.

```
db2GetRecommendationsFree( db2Version820, &recData, &sqlca ) ;
```

Typically you would combine the preceding code with a call to the snapshot APIs to take a health snapshot because recommendations are generally queried when you detect a health indicator has entered an alert state.

### **Resolving health monitor alerts using the Health Center:**

The Health Center provides support to retrieve and implement recommended actions for alert conditions.

To resolve health monitor alerts using the Health Center:

1. From the Health Center alerts view, right-click the row of the alert that you want to resolve and select Recommendation Advisor from the pop-up menu. The Recommendation Advisor opens and displays the details of the alert in a format similar to the Details window.
2. Follow the steps of the Recommendation Advisor to select the most appropriate recommendation. The Recommendation Advisor provides the functionality to implement the recommendation.

There are two types of recommendations: investigation and recommendation. The following four types of actions are supported in the Recommendation Advisor for these recommendation types:

#### **Launching a graphical administration tool**

This option will launch a graphical tool that will resolve or investigate the alert condition. The tool is launched in the context of the object against which the alert occurred.

#### **Updating configuration parameters**

The configuration parameters requiring updates are listed with current and suggested values. The suggested value can be updated as needed.

#### **Running a DB2 command script**

The recommendation action might require more than a single command. DB2 command scripts allow for multiple commands to be run to resolve the alert condition. For example, the Reorganization Required health indicator provides a DB2 command script action to run the utility.

#### **Implementing an alternative resolution**

If the action cannot be accomplished within the DB2 administration toolset, instructions are provided to resolve the alert condition using alternate methods.

**Health indicator configuration:** A default health monitor configuration is provided during installation. This ensures that the health monitor can evaluate the health of the database environment as soon as DB2 is started. However, the health monitor's behavior in evaluating health indicators and reacting to alert states can be fine-tuned through configuration for a specific user's environment.

There are different levels at which the configuration can be defined. A default configuration of factory settings is provided for each health indicator when DB2 is installed. When the health monitor starts for the first time, a copy of the factory settings provides the defaults for the instance and global settings.

Instance settings apply to the instance. Global settings apply to objects such as databases, table spaces, and table space containers in the instance that do not have customized settings defined.

Updating health indicator settings for a specific database, table space, or table space container creates object settings for the updated health indicators. The default for object settings is the global settings.

The health monitor checks the object settings when it processes a health indicator for a particular database, table space, or table space container. If the settings for a particular health indicator have never been updated, the default global settings are used to process the health indicator. The instance settings are used when the health monitor processes a health indicator for the instance.

You can alter health monitor behavior by using a number of attributes that can be configured for each health indicator. The first set of parameters (evaluation flag, thresholds, sensitivity) defines when the health monitor will generate an alert for a health indicator. The second set of parameters (action flag, actions) defines what the health monitor does upon generating the alert.

#### **Evaluation flag**

Each health indicator has an evaluation flag to enable or disable evaluation of alert state.

#### **Warning and alarm thresholds**

Threshold-based health indicators have settings defining the warning and alarm regions for the health indicator value. These warning and alarm threshold values can be modified for your particular database environment.

#### **Sensitivity parameter**

The sensitivity parameter defines the minimum amount of time, in seconds, that the health indicator value has to be in an alert state before the alert is generated. The wait time associated with the sensitivity value starts on the first refresh interval during which the health indicator value enters an alert state. You can use this value to eliminate erroneous alerts generated due to temporary spikes in resource usage.

Consider an example using the Log Utilization (*db.log\_util*) health indicator. Suppose that you review the DB2 notify log on a weekly basis. In the first week, an entry for *db.log\_util* is in alarm state. You recall having received notification for this situation, but on checking for the alert situation from the CLP, the health indicator was back in normal state. After a second week, you notice a second alarm notification entry for the same health indicator at the same time of the week. You investigate activity in your database environment on the two occasions that alerts were generated, and you discover that an application that takes a long time to commit is run weekly. This application causes the log utilization to spike for a short time, approximately eight to nine minutes, until the application commits. You can see from the history entries in the alarm notification record in the notification log, that the *db.log\_util* health indicator is evaluated every 10 minutes. Because the alert is being generated, the application time must be spanning that refresh interval. You set the sensitivity for the *db.log\_util* parameter to ten minutes. Now every time the value of *db.log\_util* first enters the warning or alarm threshold regions, the value must remain in that region for at least ten minutes before the alert is generated. No further notification entries are recorded in the notification log for this situation because the application only lasts eight to nine minutes.

#### **Action flag**

The running of actions on alert generation is controlled by the action flag. Only when the action flag is enabled will configured alert actions be run.

## Actions

Script or task actions can be configured to run on alert occurrence. For threshold-based health indicators, actions can be configured to run on warning or alarm thresholds. For state-based health indicators, actions can be configured to run on any of the possible non-normal conditions. The DB2 administration server must be running for actions to run.

The following input parameters are passed to every operating system command script:

- <health indicator shortname>
- <object name>
- <value | state>
- <alert type>

Script actions use the default interpreter on the operating system. If you want to use a non-default interpreter, create a task in the Task Center with the script content. In a multipartitioned environment, the script defined in the script action must be accessible by all partitions.

The refresh interval at which the health monitor checks each health indicator cannot be configured. The recommendation actions considered by the health monitor cannot be configured.

The health monitor configuration is stored in a binary file, HealthRules.reg:

- On Windows, HealthRules.reg is stored in x:\<SQLLIB\_PATH>\<INSTANCE\_NAME>. For example, d:\sqllib\DB2.
- On UNIX, HealthRules.reg is stored in ~/<SQLLIB\_PATH>/cfg. For example, ~/home/sqllib/cfg.

It is possible to replicate a health monitor configuration to other DB2 Version 8 instances on a Linux, UNIX, or Windows server. You can accomplish this replication by copying the binary configuration file to the appropriate directory location on the target instance.

*Retrieving health indicator configuration using the CLP:*

The GET ALERT CONFIGURATION command allows you to view the factory settings and the instance, global, and object settings.

1. To view the global settings for database-level health indicators, which apply to all databases without customized settings for the health indicator, issue the following command:

```
DB2 GET ALERT CONFIGURATION FOR DATABASES
```

2. To view the global settings for database-level health indicators, which apply to all databases without customized settings for the health indicator, issue the following command:

```
DB2 GET ALERT CONFIGURATION FOR DATABASES
```

The output of each health indicator's settings indicates whether it has been changed from its default. In the following output, the global settings have not been updated; therefore, they are the same as the default factory settings. To view factory settings for database-level health indicators, issue the same command as in the preceding example with the DEFAULT keyword.

Alert Configuration

```
Indicator Name           = db.db_op_status
Default                  = Yes
```



```

Type = State-based
Sensitivity = 0
Formula = db.db_status;
Actions = Disabled
Threshold or State checking = Enabled

Indicator Name = db.sort_shrmem_util
Default = Yes
Type = Threshold-based
Warning = 70
Alarm = 85
Unit = %
Sensitivity = 0
Formula = ((db.sort_shrheap_allocated/sheapthres_shr)
*100);
Actions = Disabled
Threshold or State checking = Enabled
...

```

3. To view the custom settings for the SAMPLE database, issue the following command:

```
DB2 GET ALERT CONFIGURATION FOR DATABASE ON SAMPLE
```

If there are no specific settings for a particular health indicator on the object specified, the global settings for all databases are displayed. To view the settings for a particular health indicator, add the `USING health-indicator-name` clause to any of the preceding examples.

*Health indicator configuration updates using the CLP:* The health indicator configuration for a particular health indicator can be updated for the global settings or the object settings for a particular object.

The `UPDATE ALERT CONFIGURATION` command has four sub-clauses that cover the different update options. Only one sub-clause can be used in each `UPDATE ALERT CONFIGURATION` command. To use more than one of the options, multiple `UPDATE ALERT CONFIGURATION` commands must be issued.

The first sub-clause, `SET parameter-name value`, provides support to update:

- The evaluation flag
- The warning and alarm thresholds (if applicable)
- The sensitivity flag
- The action flag

The parameter names for these settings are, respectively:

- `THRESHOLDSCHECKED`
- `WARNING` and `ALARM`
- `SENSITIVITY`
- `ACTIONSENABLED`

The other three sub-clauses provide support to add, to update, and to delete script or task actions.

The following commands update a threshold-based health indicator configuration for the `db.spilled_sorts` health indicator on the SAMPLE database. The update changes the warning threshold to 25, to enable actions, and to add a script action:

```

DB2 UPDATE ALERT CONFIGURATION FOR DATABASE ON SAMPLE USING DB.SPILLED_SORTS
SET WARNING 25, ACTIONSENABLED YES
DB2 UPDATE ALERT CONFIGURATION FOR DATABASE ON SAMPLE USING DB.SPILLED_SORTS
ADD ACTION SCRIPT c:\myscript TYPE OS COMMAND LINE PARAMETERS 'space'
WORKING DIRECTORY c:\ ON ALARM USER dba1 PASSWORD dba1

```



The following commands update a state-based health indicator configuration for the `ts.ts_util` health indicator for the global settings. The update defines an action to run when any table space is in backup pending state.

```
DB2 UPDATE ALERT CONFIGURATION FOR TABLESPACES USING TS.TS_UTIL
    SET ACTIONSENABLED YES
DB2 UPDATE ALERT CONFIGURATION FOR TABLESPACES USING TS.TS_UTIL
    ADD ACTION TASK 0.1 ON ATTENTION 32 ON localhost USER dba1 PASSWORD dba1
```

This update will apply to all table spaces for the instance that do not have customized settings for this health indicator.

When adding actions to a health indication configuration, the options for the `ON condition` clause are based on the type of health indicator:

- For a threshold-based health indicator, `WARNING` and `ALARM` are valid conditions.
- For a state-based health indicator, the `ON ATTENTION state` option must be used. A valid numeric state, as defined for the health indicator, should be used. The database manager and database operational state values can be found in `sqllib\include\sqlmon.h`. The table space and table space container operational values are listed in `sqllib\include\sqlutil.h`. Note that actions cannot be executed for the database manager down state. Refer to the description of the `db2.db2_op_status` health indicator for details.

*Resetting health indicator configuration using the CLP:*

The CLP provides support for the global settings to be reset to the factory settings. The object settings for a particular object can also be reset to the custom settings for that object type.

- To reset the object settings for the `SAMPLE` database to the current global settings for databases:  
DB2 RESET ALERT CONFIGURATION FOR DATABASE ON SAMPLE
- Issue the following command to reset the global settings for databases to the factory settings:  
DB2 RESET ALERT CONFIGURATION FOR DATABASES
- To reset the configuration for a particular health indicator, add the `USING health-indicator-name` clause to any of the preceding examples.

*Configuring health indicators using a client application:*

Health monitor configuration is accessible through the `db2GetAlertCfg`, `db2UpdateAlertCfg`, and `db2ResetAlertCfg` APIs in a C or C++ application. Each of these APIs can access the factory, instance, global, and object settings.

You must have an instance attachment to access the health monitor configuration. If there is not an attachment to an instance, then a default instance attachment is created. To access the health monitor configuration of a remote instance, you must first attach to that instance.

Combinations of the `objType` and `defaultType` parameters in the `db2GetAlertCfgData` structure allow access to the various levels of health indicator configuration.

Table 47. Settings for objType and defaultType to access configuration levels

| Setting          | objType and defaultType                                                                                                                                                                                                                       |
|------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Factory settings | objType = DB2ALERTCFG_OBJTYPE_{DBM   DATABASES   TABLESPACES   CONTAINERS} and defaultType = DB2ALERTCFG_DEFAULT                                                                                                                              |
| Global settings  | objType = DB2ALERTCFG_OBJTYPE_{DBM   DATABASES   TABLESPACES   CONTAINERS} and defaultType = DB2ALERTCFG_NOT_DEFAULT<br><br>or<br><br>objType = DB2ALERTCFG_OBJTYPE_{DATABASE   TABLESPACE   CONTAINER} and defaultType = DB2ALERTCFG_DEFAULT |
| Object settings  | objType = DB2ALERTCFG_OBJTYPE_{DATABASE   TABLESPACE   CONTAINER} and defaultType = DB2ALERTCFG_NOT_DEFAULT                                                                                                                                   |

1. To get the specific object setting for health indicators on the SAMPLE database:

a. Include the db2ApiDf.h DB2 header file, found in the sqllib\include directory.

```
#include <db2ApiDf.h>
```

b. Declare and initialize the sqlca and db2GetAlertCfgData structures.

```
struct sqlca ca;
memset (&sqlca, '\0', sizeof(struct sqlca));
```

```
char* objName = NULL;
char* dbName = "SAMPLE";
db2Uint32 objType = DB2ALERTCFG_OBJTYPE_DATABASE;
db2Uint32 defaultType = DB2ALERTCFG_NOT_DEFAULT;
```

```
db2GetAlertCfgData data = {objType, objName, defaultType, dbName, 0, NULL} ;
```

c. Call the db2GetAlertCfg API.

```
rc = db2GetAlertCfg (db2Version810, &data, &ca);
```

d. Process the returned configuration and free the buffer allotted by the API.

```
if (rc >= SQL0_OK) {
    if ((data.ioNumIndicators > 0) && (data.pioIndicators != NULL)) {
        db2GetAlertCfgInd *pIndicators = data.pioIndicators;

        for (db2Uint32 i=0; i < data.ioNumIndicators; i++) {
            //process the entry as necessary using fields defined in db2ApiDf.h
        }
    }

    db2GetAlertCfgFree (db2Version810, &data, &ca);
}
```

2. The following steps detail the procedure to update the alert configuration of the **db.sort\_shrmem\_util** health indicator for the global settings for database objects, setting warning threshold to 80 and adding task action 1.1:

a. Include the db2ApiDf.h DB2 header file, found in the sqllib\include directory.

```
#include <db2ApiDf.h>
```

b. Declare and initialize the sqlca and db2AlertTaskAction structures.

```
struct sqlca ca;
memset (&sqlca, '\0', sizeof(struct sqlca));
```

```
db2Uint32 objType = DB2ALERTCFG_OBJTYPE_DATABASES;
```

```
db2Uint32 taskCondition = DB2ALERTCFG_CONDITION_WARNING;
char* taskname = "1.1";
```

- ```

char* hostname = NULL;
char* userid = "nobody";
char* password = "nothing";

db2AlertTaskAction newTask={taskname,taskCondition,userid,password,hostname};

```
- c. Declare and initialize the db2UpdateAlertCfgData structure.
- ```

struct db2UpdateAlertCfgData setData;

setData.iObjType = objType;
setData.piObjName = NULL;
setData.piDbName = NULL;

setData.iIndicatorID = 1002;

setData.iNumIndAttribUpdates = 1;
setData.piIndAttribUpdates[0].iAttribID = DB2ALERTCFG_WARNING;
setData.piIndAttribUpdates[0].piAttribValue == 80;

setData.iNumActionUpdates = 0;
setData.piActionUpdates = NULL;

setData.iNumActionDeletes = 0;
setData.piActionDeletes = NULL;

setData.iNumNewActions = 1;
setData.piNewActions[0].iActionType = DB2ALERTCFG_ACTIONTYPE_TASK;
setData.piNewActions[0].piScriptAttribs = NULL;
setData.piNewActions[0].piTaskAttribs = &newTask;

```
- d. Call the db2UpdateAlertCfg API.
- ```

rc = db2UpdateAlertCfg(db2Version810, &setData, &ca);

```

3. The following steps detail the procedure to RESET the custom settings for the MYTS table space in the SAMPLE database.

- a. Include the db2ApiDf.h DB2 header file, found in the sqllib\include directory.

```
#include <db2ApiDf.h>
```

- b. Declare and initialize the sqlca and db2ResetAlertCfgData structures.

```

struct sqlca ca;
memset (&sqlca, '\0', sizeof(struct sqlca));

```

```

char* objName = "MYTS";
char* dbName = "SAMPLE";
db2Uint32 objType = DB2ALERTCFG_OBJTYPE_TABLESPACE;

```

```
db2ResetAlertCfgData data = {objType, objName, dbName};
```

- c. Call the db2ResetAlertCfg API.

```
rc = db2ResetAlertCfg (db2Version810, &data, &ca);
```

#### *Configuring health indicators using the Health Center:*

The Health Center provides graphical interfaces to view, update, and reset health indicator configurations. The configuration for health indicators is stored in the health monitor within the instance.

To define, change, enable or disable threshold or sensitivity settings for a health indicator, and to define, change, enable or disable running tasks or scripts when a health alert occurs for a health indicator, you must have one of the following authorities:

- SYSADM
- SYSMAINT
- SYSCTRL

You can adjust the health indicator settings for an instance, the global health indicator settings for database objects contained in the instance, and for individual database objects.

1. To configure health indicators using the Health Center:
    - a. Select the instance whose health indicators you want to configure.
    - b. From the **Selected** menu, or from the right-click menu, click **Configure** then **Health Indicator Settings**. The Health Indicator Configuration Launchpad opens.
    - c. Each level of configuration settings that can be updated has a button on the launchpad. Select the button for the level of configuration that you want to view, update, or reset. Each button launches a Health Indicator Configuration window at the chosen level of configuration settings.
    - d. To update the health indicator settings, select the row of the health indicator in the Current health indicator settings table.
    - e. From the **Select** menu, or from the right-click menu, select **Edit**. The Configure Health Indicator notebook opens and displays the following information:
      - A description of the health indicator is provided by clicking **Tell Me More**.
      - The evaluation of the health indicator can be enabled and disabled using the **Evaluate** check box.

**Note:** The **Evaluate** flag can also be disabled from the Health Center Alerts View for current alerts through the right-click menu option on a current alert. This option will disable the evaluation of the health indicator on the next refresh of the indicator in the health monitor. When selecting **Disable evaluation** for an alert in the Health Center, the evaluation flag is set to false for the health indicator, but the alert will not be removed from the Alerts view until the following events take place:

    - The health monitor refresh interval for that particular health indicator is reached
    - The health monitor refreshes the health indicator evaluation
    - The Health Center refreshes its view of status  - On the Alert page, for threshold-based health indicators, the warning and alarm thresholds can be updated. The sensitivity for any health indicator can also be set on this page.
  - On the Actions page, a task or script action can be selected to run when an alert occurs. Actions can be configured to run on warning or alarm conditions for threshold-based health indicators or on any non-normal condition for state-based health indicators. You can enable or disable the execution of actions by selecting or deselecting the **Enable actions** check box. To add, update, or remove task or script actions, use the buttons beside the **Script actions** and **Task actions** tables.
2. To view the factory health indicator settings for the instance:
  - a. In the Health Indicator Configuration launchpad, click **Instance Settings**.
  - b. In the Instance Health Indicator Configuration window, click **View Default**.
3. To view the global health indicator settings for databases, table spaces, or table space containers:
  - a. In the Health Indicator Configuration launchpad, click **Global Settings**.
  - b. Select the object type in the Global Health Indicator Configuration window.
  - c. To view the factory defaults for these global settings, click **View Default**

4. To view the health indicator settings for a database object:
  - a. In the Health Indicator Configuration launchpad, click **Object Settings**.
  - b. Select the object in the Object Health Indicator Configuration window.
  - c. To view the global default health indicator settings for this object type, click **View Default**.

In each of these windows, to reset the settings of all the displayed health indicators to their defaults, click **Reset to Default**. You can also reset individual health indicators by right-clicking one or more health indicators that you want in the **Current health indicator settings** field and selecting **Reset to Default** from the pop-up menu.

*Health monitor alert actions on combined states:*

Alert actions are tasks or scripts that are run when a health indicator goes into an alert state.

Starting in DB2 V9.1, the health monitor alert actions defined for the health indicator **ts.ts\_op\_status** on a single alert state are executed whenever this state is set for the table space, irrespective of the other combined states. This makes it possible to run alert actions on a specific table space state even when it is set in conjunction with other states.

In the following example, an alert action script1 defined on attention state QUIESCED:share will run, even if the table space state is QUIESCED:share and QUIESCE:update at the same time.

```
db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status
set actionsenabled yes')
db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status
add action script /home/guest001/script1 type operating system command line
parameters userParam working directory /home/guest001/
on attention QUIESCED_SHARE on aix1 user guest001 using passwd')
```

In the following example, an alert action defined using a combination of states ( QUIESCED:share + QUIESCED:update = 3 ) is executed if and only if the table space state is both QUIESCED:share and QUIESCED:update.

```
db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status
set actionsenabled yes')
db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status
add action script /home/guest001/script1 type operating system command line
parameters userParam working directory /home/guest001/
on attention 3 on aix1 user guest001 using passwd')
```

Starting in DB2 V9.1, health monitor alert actions defined on an object with the same action attributes (name, working directory, command line parameters, host, user and password) run only once, even if it was defined on multiple alert states.

In the following example, the same action is defined on two different alert states. The action is only executed once for a given table space, even if the table space state is in both QUIESCED:share and QUIESCED:update.

```
db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status
add action script /home/guest001/script1 type operating system command line
parameters userParam working directory /home/guest001/
on attention QUIESCED_SHARE on aix1 user guest001 using passwd')
```

```
db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status
add action script /home/guest001/script1 type operating system command line
parameters userParam working directory /home/guest001/
on attention QUIESCED_UPDATE on aix1 user guest001 using passw0rd')
```

## Health indicators

The health monitor uses health indicators to evaluate the health of specific aspects of database manager performance or database performance. A health indicator measures the health of some aspect of a particular class of database objects, such as table spaces. Criteria are applied to the measurement to determine healthiness. The criteria applied depends on the type of health indicator. A determination of unhealthiness is based on the criteria generates an alert.

Three types of health indicators are returned by the health monitor:

- **Threshold-based** indicators are measurements that represent a statistic (on a continuous range of values) of the behavior of the object. Warning and alarm threshold values define boundaries or zones for normal, warning, and alarm ranges. Threshold-based health indicators have three valid states: Normal, Warning, or Alarm.
- **State-based** indicators are measurements that represent a finite set of two or more distinct states of an object that defines whether the database object or resource is operating normally. One of the states is normal and all others are considered non-normal. State-based health indicators have two valid states: Normal, Attention.
- **Collection state-based** indicators are database-level measurements that represent an aggregate state or one or more objects within the database. Data is captured for each object in the collection and the highest severity of conditions among those objects is represented in the aggregated state. If one or more objects in the collection are in a state requiring an alert, the health indicator shows Attention state. Collection state-based health indicators have two valid states: Normal, Attention.

Health indicators exist at the instance, database, table space, and table space container level.

You can access health monitor information through the Health Center, the CLP, or APIs. You can configure health indicators through these same tools.

An alert is generated in response to either a change from a normal to a non-normal state or a change in the health indicator value to a warning or alarm zone that is based on defined threshold boundaries. There are three types of alerts: attention, warning, and alarm.

- For health indicators measuring distinct states, an attention alert is issued if a non-normal state is registered.
- For health indicators measuring a continuous range of values, threshold values define boundaries or zones for normal, warning and alarm states. For example, if the value enters the threshold range of values that defines an alarm zone, an alarm alert is issued to indicate that the problem needs immediate attention.

The health monitor will only send notification and run an action on the first occurrence of a particular alert condition for a given health indicator. If the health indicator stays in a particular alert condition, no further notification will be sent and no further actions will be run. If the health indicator changes alert conditions, or goes back to normal state and re-enters the alert condition, notification will be sent anew and actions will be run.

The following table shows an example of a health indicator at different refresh intervals and the health monitor response to the health indicator state. This example uses the default warning of 80% and alarm thresholds of 90%.

Table 48. Health indicator conditions at different refresh intervals

Refresh interval	Value of ts.ts_util (Table space utilization) health indicator	State of ts.ts_util health indicator	Health monitor response
1	80	warning	notification of warning is sent, actions for a warning alert condition are run
2	81	warning	no notification is sent, no actions are run
3	75	normal	no notification is sent, no actions are run
4	85	warning	notification of warning is sent, actions for a warning alert condition are run
5	90	alarm	notification of alarm is sent, actions for an alarm condition are run

## Health monitor interface mappings to logical data groups

The following table lists all the supported health snapshot request types.

Table 49. Health monitor interface mappings to logical data groups

API request type	CLP command	SQL table function	Logical data groups
SQLMA_DB2	get health snapshot for dbm	HEALTH_DBM_INFO	db2
		HEALTH_DBM_HI	health_indicator
	get health snapshot for dbm show detail	HEALTH_DBM_HI_HIS	health_indicator_history
SQLMA_DBASE	get health snapshot for database on <i>dbname</i>	HEALTH_DB_INFO	dbase
		HEALTH_DB_HI	health_indicator
	get health snapshot for database on <i>dbname</i> show detail	HEALTH_DB_HI_HIS	health_indicator_history
SQLMA_DBASE with SQLM_HMON_OPT_COLL_FULL in the agent_id	get health snapshot for database on <i>dbname</i> with full collection	HEALTH_DB_HIC	health_indicator, hi_obj_list
		get health snapshot for database on <i>dbname</i> show detail with full collection	HEALTH_DB_HIC_HIST
SQLMA_DBASE_ALL	get health snapshot for all databases	HEALTH_DB_INFO	dbase
		HEALTH_DB_HI	health_indicator
	get health snapshot for all databases show detail	HEALTH_DB_HI_HIS	health_indicator_history



Table 49. Health monitor interface mappings to logical data groups (continued)

API request type	CLP command	SQL table function	Logical data groups
SQLMA_DBASE_TABLESPACES	get health snapshot for tablespaces on <i>dbname</i>	HEALTH_TS_INFO	tablespace
		HEALTH_TS_HI	health_indicator
		HEALTH_CONT_INFO	tablespace_container
		HEALTH_CONT_HI	health_indicator
	get health snapshot for tablespaces on <i>dbname</i> show detail	HEALTH_TS_HI_HIS	health_indicator_history
		HEALTH_CONT_HI_HIS	health_indicator_history

The following figure shows the order that logical data groupings can appear in a health snapshot data stream.

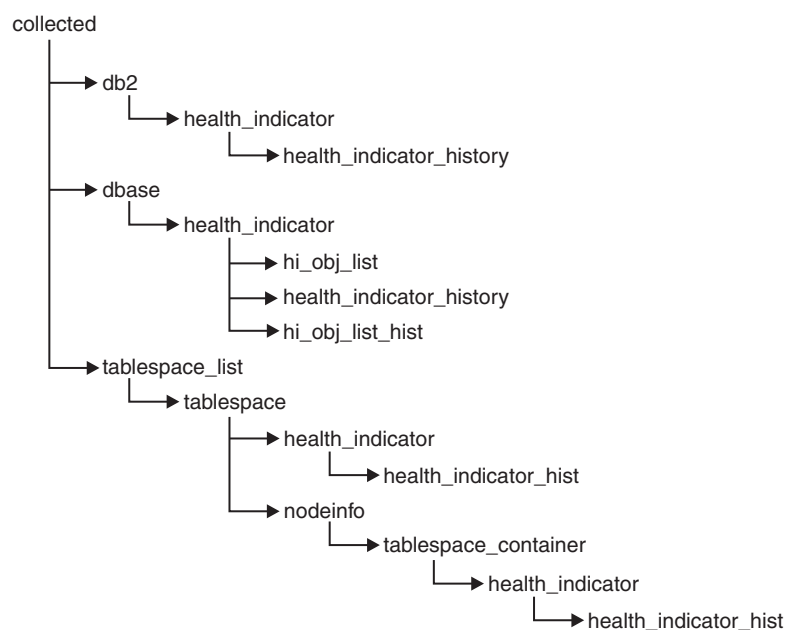


Figure 6. Health snapshot logical data groupings

## Health indicators summary

The following tables list all health indicators, grouped by category.

Table 50. Database automatic storage utilization health indicators

Name	Identifier	Additional Information
Database Automatic Storage Utilization	db.auto_storage_util	"db.auto_storage_util - Database automatic storage utilization health indicator" on page 165

Table 51. Table space storage health indicators

Name	Identifier	Additional Information
Table Space Automatic Resize Status	ts.ts_auto_resize_status	"ts.ts_auto_resize_status - Table space automatic resize status health indicator" on page 166



Table 51. Table space storage health indicators (continued)

Name	Identifier	Additional Information
Automatic Resize Table Space Utilization	ts.ts_util_auto_resize	"ts.ts_util_auto_resize - Automatic resize table space utilization health indicator" on page 166
Table Space Utilization	ts.ts_util	"ts.ts_util - Table Space Utilization" on page 167
Table Space Container Utilization	tsc.tscont_util	"tsc.tscont_util - Table Space Container Utilization" on page 168
Table Space Operational State	ts.ts_op_status	"ts.ts_op_status - Table Space Operational State" on page 169
Table Space Container Operational State	tsc.tscont_op_status	"tsc.tscont_op_status - Table Space Container Operational State" on page 169
Table Space Automatic Resize Status	ts.ts_auto_resize_status	"ts.ts_auto_resize_status - Table space automatic resize status health indicator" on page 166

Table 52. Sorting health indicators

Name	Identifier	Additional Information
Private Sort Memory Utilization	db2.sort_privmem_util	"db2.sort_privmem_util - Private Sort Memory Utilization" on page 169
Shared Sort Memory Utilization	db.sort_shrmem_util	"db.sort_shrmem_util - Shared Sort Memory Utilization" on page 170
Percentage of Sorts That Overflowed	db.spilled_sorts	"db.spilled_sorts - Percentage of Sorts That Overflowed" on page 170
Long Term Shared Sort Memory Utilization	db.max_sort_shrmem_util	"db.max_sort_shrmem_util - Long Term Shared Sort Memory Utilization" on page 171

Table 53. Database manager health indicators

Name	Identifier	Additional Information
Instance Operational State	db2.db2_op_status	"db2.db2_op_status - Instance Operational State" on page 172
Instance Highest Severity Alert State	-	"Instance Highest Severity Alert State" on page 172

Table 54. Database health indicators

Name	Identifier	Additional Information
Database Operational State	db.db_op_status	"db.db_op_status - Database Operational State" on page 173
Database Highest Severity Alert State	-	"Database Highest Severity Alert State" on page 173

Table 55. Maintenance health indicators

Name	Identifier	Additional Information
Reorganization Required	db.tb_reorg_req	"db.tb_reorg_req - Reorganization Required" on page 173

Table 55. Maintenance health indicators (continued)

Name	Identifier	Additional Information
Statistics Collection Required health indicator	db.tb_runstats_req	"db.tb_runstats_req - Statistics Collection Required" on page 174
Database Backup Required	db.db_backup_req	"db.db_backup_req - Database Backup Required" on page 174

Table 56. High availability disaster recovery health indicators

Name	Identifier	Additional Information
HADR Operational Status health indicator	db.hadr_op_status	"db.hadr_op_status - HADR Operational Status" on page 175
HADR Log Delay health indicator	db.hadr_delay	"db.hadr_delay - HADR Log Delay" on page 175

Table 57. Logging health indicators

Name	Identifier	Additional Information
Log Utilization	db.log_util	"db.log_util - Log Utilization" on page 176
Log File System Utilization	db.log_fs_util	"db.log_fs_util - Log File System Utilization" on page 176

Table 58. Application concurrency health indicators

Name	Identifier	Additional Information
Deadlock Rate	db.deadlock_rate	"db.deadlock_rate - Deadlock Rate" on page 177
Lock List Utilization	db.locklist_util	"db.locklist_util - Lock List Utilization" on page 177
Lock Escalation Rate	db.lock_escal_rate	"db.lock_escal_rate - Lock Escalation Rate" on page 178
Percentage of Applications Waiting on Locks	db.apps_waiting_locks	"db.apps_waiting_locks - Percentage of Applications Waiting on Locks" on page 179

Table 59. Package cache, catalog cache, and workspace health indicators

Name	Identifier	Additional Information
Catalog Cache Hit Ratio	db.catcache_hitratio	"db.catcache_hitratio - Catalog Cache Hit Ratio" on page 179
Package Cache Hit Ratio	db.pkgcache_hitratio	"db.pkgcache_hitratio - Package Cache Hit Ratio" on page 180
Shared Workspace Hit Ratio	db.shrworkspace_hitratio	"db.shrworkspace_hitratio - Shared Workspace Hit Ratio" on page 180

Table 60. Memory health indicators

Name	Identifier	Additional Information
Monitor Heap Utilization	db2.mon_heap_util	"db2.mon_heap_util - Monitor Heap Utilization" on page 180
Database Heap Utilization	db.db_heap_util	"db.db_heap_util - Database Heap Utilization" on page 181

Table 61. Federated health indicators

Name	Identifier	Additional Information
Nickname Status	db.fed_nicknames_op_status	"db.fed_nicknames_op_status - Nickname Status" on page 181
Data Source Server Status	db.fed_servers_op_status	"db.fed_servers_op_status - Data Source Server Status" on page 182

**Health indicator format:**

A description of the data collected by the health indicator.

The documentation for health indicators is described in a standard format as follows:

**Identifier**

The name of the health indicator. This identifier is used for configuration from the CLP.

**Health monitor level**

The level at which the health indicator is captured by the health monitor.

**Category**

The category for the health indicator.

**Type** The type of the health indicator. There are four possible values for type:

- Upper-bounded threshold-based, where the progression to an alert is: Normal, Warning, Alarm
- Lower-bounded threshold-based
- State-based, where one state is normal and all others are non-normal
- Collection state-based, where the state is based on the aggregation of states from objects in the collection

**Unit** The unit of the data measured in the health indicator, such as percentage. This is not applicable for state-based or collection state-based health indicators.

**Table space storage health indicators:**

*Health indicators for DMS table spaces:*

This table describes which table space health indicators are relevant for a DMS table space based on the characteristics of the table space:

Table 62. Relevant table space health indicators for a DMS table space

Table space characteristics	Maximum table space size defined	Maximum table space size undefined
Automatic resize enabled = Yes	<p>ts.ts_util_auto_resize - Tracks percentage of table space used relative to the maximum defined by you. An alert indicates that the table space will soon be full and requires intervention by you. As long as the maximum size has been set to a reasonable value (that is, the amount of space specified by the maximum size does exist), this is the most important health indicator for this configuration.</p> <p>ts.ts_util - Tracks usage of currently allocated table space storage. An alert may not require intervention by you to resolve any problems since the table space will attempt to increase in size when it is full.</p> <p>ts.ts_auto_resize_status - Tracks health of resize attempts. An alert indicates that the table space failed to resize (that is, the table space is full).</p>	<p>ts.ts_util_auto_resize - Not applicable. No upper bound specified for the table space size.</p> <p>ts.ts_util - Tracks usage of currently allocated table space storage. An alert may not require intervention by you to resolve any problems since the table space will attempt to increase in size.</p> <p>ts.ts_auto_resize_status - Tracks health of resize attempts. An alert indicates that the table space failed to resize (that is, the table space is full). <b>Note:</b> If a DMS table space is defined using automatic storage and there is no maximum size specified, you should also pay attention to the db.auto_storage_util health indicator. This health indicator tracks utilization of the space associated with the database storage paths. When this space fills up, the table space is unable to grow. This may result in a table space full condition.</p>
Automatic resize enabled = No	Not a valid configuration. Maximum table space size is only valid for table spaces that have automatic resize enabled.	<p>ts.ts_util_auto_resize - Not applicable. Table space will not attempt to resize.</p> <p>ts.ts_util - Tracks usage of currently allocated table space storage. An alert indicates a table space full condition and requires immediate intervention by you. The table space will not attempt to resize itself.</p> <p>ts.ts_auto_resize_status - Not applicable. Table space will not attempt to resize.</p>

*db.auto\_storage\_util - Database automatic storage utilization health indicator:*

This health indicator tracks the consumption of storage for the defined database storage paths.

**Identifier**

db.auto\_storage\_util

**Health monitor level**

Database

**Category**

Database

**Type** Upper-bounded threshold-based

**Unit** Percentage

When automatic storage table spaces are created, containers are allocated automatically for these table spaces on the database storage paths. If there is no more space on any of the file systems on which the database storage paths are defined, automatic storage table spaces will be unable to increase in size and may become full.

The indicator is calculated using the formula:

$$(db.auto\_storage\_used / db.auto\_storage\_total) * 100$$

where

- *db.auto\_storage\_used* is the sum of used space across all physical file systems identified in the list of database storage paths
- *db.auto\_storage\_total* is the sum of total space across all physical file systems identified in the list of database storage paths

Database automatic storage path utilization is measured as a percentage of the space consumed on the database storage path file systems, where a high percentage indicates less than optimal function for this indicator.

The calculation of time remaining to fullness in the additional information is a prediction of how much time is remaining until all free space is consumed.

*ts.ts\_auto\_resize\_status* - Table space automatic resize status health indicator:

This health indicator identifies whether table space resize operations are succeeding for DMS table spaces which have automatic resize enabled. When a DMS table space with automatic resize enabled fails to increase in size, it is effectively full. This condition may be due to lack of free space on the file systems on which the table space containers are defined, or a result of the table space automatic resize settings. For example, the defined maximum size may have been reached, or the increase amount may be set too high to be accommodated by the remaining free space.

**Identifier**

ts.ts\_auto\_resize\_status

**Health monitor level**

Table Space

**Category**

Table Space Storage

**Type** State-based

**Unit** Not applicable

*ts.ts\_util\_auto\_resize* - Automatic resize table space utilization health indicator:

This health indicator tracks the consumption of storage for each automatic resize DMS table space on which a maximum size has been defined. The DMS table space is considered full when the maximum size has been reached.

**Identifier**

ts.ts\_util\_auto\_resize

**Health monitor level**

Table Space

**Category**

Table Space Storage

**Type** Upper-bounded threshold-based**Unit** Percentage

The indicator is calculated using the formula:

$$((ts.used * ts.page\_size) / ts.max\_size) * 100$$

where

- *ts.used* is the value of “tablespace\_used\_pages - Used pages in table space monitor element” on page 697
- *ts.page\_size* is the value of “tablespace\_page\_size - Table space page size monitor element” on page 689
- *ts.max\_size* is the value of “tablespace\_max\_size - Maximum table space size” on page 686

Automatic resize DMS table space utilization is measured as a percentage of the maximum table space storage consumed. A high percentage indicates the table space is approaching fullness. The short term and long term growth rates, included in the additional information for this indicator, can be used to determine if the current rate of growth is a short term aberration, or consistent with long term growth.

The calculation of time remaining to fullness in the additional information is a prediction of how much time is remaining until the maximum size has been reached.

*ts.ts\_util* - Table Space Utilization:

This health indicator tracks the consumption of storage for each DMS table space.

**Identifier**

ts.ts\_util

**Health monitor level**

Table Space

**Category**

Table Space Storage

**Type** Upper-bounded threshold-based**Unit** Percentage

The DMS table space is considered full when all containers are full.

If automatic resize is enabled on the table space, this health indicator will not be evaluated. Instead, the database automatic storage utilization **db.auto\_storage\_util** and table space automatic resize status (**ts.ts\_auto\_resize\_status**) health indicators are relevant for table space storage monitoring. The automatic resize table space utilization (**ts.ts\_util\_auto\_resize**) health indicator will also be available if a maximum size was defined on this table space. The table space utilization percentage can still be retrieved from column TBSP\_UTILIZATION\_PERCENT of the TBSP\_UTILIZATION administrative view if it is required.

The indicator is calculated using the formula:

$(ts.used / ts.usable) * 100$

where

- *ts.used* is the value of “tablespace\_used\_pages - Used pages in table space monitor element” on page 697
- *ts.usable* is the value of “tablespace\_usable\_pages - Usable pages in table space monitor element” on page 697

Table space utilization is measured as the percentage of space consumed, where a high percentage indicates less than optimal function for this indicator.

The short term and long term growth rates, included in the additional information for this indicator, can be used to determine if current rate of growth is a short term aberration or consistent with longer term growth.

The calculation of time remaining to fullness in the additional information is a prediction of how much time is remaining until all free space is consumed.

*tsc.tscont\_util* - Table Space Container Utilization:

This health indicator tracks the consumption of storage for each SMS table space that is not using automatic storage.

**Identifier**

tsc.tscont\_util

**Health monitor level**

Table Space Container

**Category**

Table Space Storage

**Type** Upper-bounded threshold-based

**Unit** Percentage

An SMS table space is considered full if there is no more space on any of the file systems for which containers are defined.

If free space is not available on the file system to expand an SMS container, the associated table space becomes full.

An alert may be issued for each container defined on the file system that is running out of free space.

The indicator is calculated using the formula:

$(fs.used / fs.total) * 100$

where fs is the file system in which the container resides.

SMS table space utilization is measured as the percentage of space consumed, where a high percentage indicates less than optimal function for this indicator.

The short term and long term growth rates, included in the additional information for this indicator, can be used to determine if current rate of growth is a short term aberration or consistent with longer term growth.

The calculation of time remaining to fullness in the additional information is a prediction of how much time is remaining until all free space is consumed.

*ts.ts\_op\_status* - Table Space Operational State:

The state of a table space can restrict activity or tasks that can be performed. A change from normal to another state may generate an Attention alert.

**Identifier**

ts.ts\_op\_status

**Health monitor level**

Table Space

**Category**

Table Space Storage

**Type** State-based

**Unit** Not applicable

*tsc.tscont\_op\_status* - Table Space Container Operational State:

This health indicator tracks the accessibility of the table space container. The accessibility of the container can restrict activity or tasks that can be performed. If the container is not accessible, an Attention alert may be generated.

**Identifier**

tsc.tscont\_op\_status

**Health monitor level**

Table Space Container

**Category**

Table Space Storage

**Type** State-based

**Unit** Not applicable

**Sorting health indicators:**

*db2.sort\_privmem\_util* - Private Sort Memory Utilization:

This indicator tracks the utilization of the private sort memory. If `db2.sort_heap_allocated` (system monitor element)  $\geq$  `sheapthres` (DBM configuration parameter), sorts may not be getting full sort heap as defined by the `sortheap` parameter and an alert may be generated.

**Identifier**

db2.sort\_privmem\_util

**Health monitor level**

Database

**Category**

Sorting

**Type** Upper-bounded threshold-based

**Unit** Percentage

Sorting is considered healthy if there is sufficient heap space in which to perform sorting and sorts do not overflow unnecessarily.



The indicator is calculated using the formula:

$(db2.sort\_heap\_allocated / sheapthres)*100$

The Post Threshold Sorts snapshot monitor element measures the number of sorts that have requested heaps after the sort heap threshold has been exceeded. The value of this indicator, shown in the Additional Details, indicates the degree of severity of the problem for this health indicator.

The Maximum Private Sort Memory Used snapshot monitor element maintains a private sort memory high watermark for the instance. The value of this indicator, shown in the Additional Information, indicates the maximum amount of private sort memory that has been in use at any one point in time since the instance was last recycled. This value can be used to help determine an appropriate value for *sheapthres*.

*db.sort\_shrmem\_util* - Shared Sort Memory Utilization:

This indicator tracks the utilization of the shared sort memory. The *sheapthres\_shr* database configuration parameter is a hard limit. If the allocation is close to the limit, an alert may be generated.

**Identifier**

db.sort\_shrmem\_util

**Health monitor level**

Database

**Category**

Sorting

**Type** Upper-bounded threshold-based

**Unit** Percentage

Sorting is considered healthy if there is sufficient heap space in which to perform sorting and sorts do not overflow unnecessarily.

The indicator is calculated using the formula:

$(db.sort\_shrheap\_allocated / sheapthres\_shr)*100$

Note that if *sheapthres\_shr* is set to 0, then *sheapthres* serves as the shared sort heap threshold.

The Maximum Shared Sort Memory Used snapshot monitor element maintains a shared sort memory high watermark for the database. The value of this indicator, shown in the Additional Information, indicates the maximum amount of shared sort memory that has been in use at any one point in time since the database has been active. This value can be used to help determine an appropriate value for the shared sort memory threshold.

Consider using the self-tuning memory feature to have sort memory resources automatically allocated as required by the current workload. If you have the self tuning memory feature enabled for the sort memory area, you should configure this health indicator to disable threshold checking.

*db.spilled\_sorts* - Percentage of Sorts That Overflowed:

Sorts that overflow to disk can cause significant performance degradation. If this occurs, an alert may be generated.

**Identifier**

db.spilled\_sorts

**Health monitor level**

Database

**Category**

Sorting

**Type** Upper-bounded threshold-based

**Unit** Percentage

Sorting is considered healthy if there is sufficient heap space in which to perform sorting and sorts do not overflow unnecessarily.

The indicator is calculated using the formula:

$$\frac{(\text{db.sort\_overflows}_t - \text{db.sort\_overflows}_{t-1})}{(\text{db.total\_sorts}_t - \text{db.total\_sorts}_{t-1}) * 100}$$

where  $t$  is the current snapshot and  $t-1$  is a snapshot 1 hour ago. The system monitor element db.sort\_overflows (based on the sort\_overflows monitor element) is the total number of sorts that ran out of sort heap and may have required disk space for temporary storage. The element db.total\_sorts (based on the total\_sorts monitor element) is the total number of sorts that have been executed.

Consider using the self-tuning memory feature to have sort memory resources automatically allocated as required by the current workload. If you have the self tuning memory feature enabled for the sort memory area, you should configure this health indicator to disable threshold checking.

*db.max\_sort\_shrmem\_util - Long Term Shared Sort Memory Utilization:*

This indicator tracks an over-configured shared sort heap, looking to see if there are resources that can be freed for use somewhere else in the DB2 database system.

**Identifier**

db.max\_sort\_shrmem\_util

**Health monitor level**

Database

**Category**

Sorting

**Type** Lower-bounded threshold-based

**Unit** Percentage

Sorting is considered healthy if there is sufficient heap space in which to perform sorting and if sorts do not overflow unnecessarily.

An alert might be generated when the percentage usage is low.

The indicator is calculated using the formula:

$$(\text{db.max\_shr\_sort\_mem} / \text{sheapthres\_shr}) * 100$$

The system monitor element `db.max_shr_sort_mem` (based on the `sort_shrheap_top` monitor element) is the high watermark for shared sort memory usage.

Consider using the self-tuning memory feature to have sort memory resources automatically allocated as required by the current workload. If you have the self-tuning memory feature enabled for the sort memory area, you should configure this health indicator to disable threshold checking.

#### **Database manager (DBMS) health indicators:**

*db2.db2\_op\_status - Instance Operational State:*

An instance is considered healthy if the instance state does not restrict activity or tasks being performed.

##### **Identifier**

`db2.db2_op_status`

##### **Health monitor level**

Instance

##### **Category**

DBMS

**Type** State-based

**Unit** Not applicable

The state can be one of the following: Active, Quiesce pending, Quiesced, or Down. A non-Active state may generate an Attention alert.

The health monitor is unable to execute actions for the `db2.db2_op_status` health indicator if the indicator enters the down state. This state can arise, for example, when an instance that the indicator is monitoring becomes inactive because of an explicit stop request or an abnormal termination. If you want to have the instance restart automatically after any abnormal termination, you can configure the fault monitor (`db2fm`) to keep the instance highly available.

*Instance Highest Severity Alert State:*

This indicator represents the rolled-up alert state of an instance being monitored. The alert state of an instance is the highest alert state of the instance and its databases, and database objects being monitored.

##### **Identifier**

Not applicable. This health indicator does not have configuration or recommendations support.

##### **Health monitor level**

Instance

##### **Category**

DBMS

**Type** State-based

**Unit** Not applicable

The order of the alert states is as follows:

- Alarm
- Warning

- Attention
- Normal

The alert state of the instance determines the overall health of the DB2 database system.

**Database health indicators:**

*db.db\_op\_status - Database Operational State:*

The state of the database can restrict activity or tasks that can be performed. The state can be one of the following: Active, Quiesce pending, Quiesced, or Rollforward. A change from Active to another state may generate an Attention alert.

**Identifier**

db.db\_op\_status

**Health monitor level**

Database

**Category**

Database

**Type** State-based

**Unit** Not applicable

*Database Highest Severity Alert State:*

This indicator represents the rolled-up alert state of the database being monitored. The alert state of a database is the highest alert state of the database and its objects.

**Identifier**

Not applicable. This health indicator does not have configuration or recommendations support.

**Health monitor level**

Database

**Category**

Database

**Type** State-based

**Unit** Not applicable

The order of the alert states is as follows:

- Alarm
- Warning
- Attention
- Normal

**Maintenance health indicators:**

*db.tb\_reorg\_req - Reorganization Required:*

This health indicator tracks the need to reorganize tables or indexes within a database. Tables or all indexes defined on a table require reorganization to

eliminate fragmented data. The reorganization is accomplished by compacting the information and reconstructing the rows or index data. The result might yield an improved performance and freed space in the table or indexes.

**Identifier**

db.tb\_reorg\_req

**Health monitor level**

Database

**Category**

Database Maintenance

**Type** Collection state-based

**Unit** Not applicable

You can filter the set of tables evaluated by this health indicator by specifying in your automatic maintenance policy the names of the tables to be evaluated. This can be done using the Automatic Maintenance wizard.

An attention alert might be generated to indicate that reorganization is required. Reorganization can be automated by setting the AUTO\_REORG database configuration parameter to ON. If automatic reorganization is enabled, the attention alert indicates either that one or more automatic reorganizations could not complete successfully or that there are tables which require reorganization, but automatic reorganization is not being performed because the size of the table per database partition exceeds the maximum size criteria for tables that should be considered for offline reorganization. Refer to the collection details of this health indicator for the list of objects that need attention.

*db.tb\_runstats\_req - Statistics Collection Required:*

This health indicator tracks the need to collect statistics for tables and their indexes within a database. Tables and all indexes defined on a table require statistics to improve query execution time.

**Identifier**

db.tb\_runstats\_req

**Health monitor level**

Database

**Category**

Database Maintenance

**Type** Collection state-based

**Unit** Not applicable

The tables considered by this health indicator can be limited using an SQL query. The scope in the additional information displays the subselect clause on system tables for this query.

An attention alert may be generated to indicate that statistics collection is required. Statistics can be automatically collected by setting the AUTO\_RUNSTATS database configuration parameter to ON. If automatic statistics collection is enabled, the attention alert indicates that one or more automatic statistics collection actions did not complete successfully.

*db.db\_backup\_req - Database Backup Required:*

This health indicator tracks the need for a backup on the database. Backups should be taken regularly as part of a recovery strategy to protect your data against the possibility of loss in the event of a hardware or software failure.

**Identifier**

db.db\_backup\_req

**Health monitor level**

Database

**Category**

Database Maintenance

**Type** State-based

**Unit** Not applicable

This health indicator determines when a database backup is required based on the time elapsed and amount of data changed since the last backup.

An attention alert might be generated to indicate that a database backup is required. Database backups can be automated by setting the AUTO\_DB\_BACKUP database configuration parameter to ON. If automatic database backups are enabled, the attention alert indicates that one or more automatic database backups did not complete successfully.

**High availability disaster recovery (HADR) health indicators:**

*db.hadr\_op\_status - HADR Operational Status:*

This health indicator tracks the high availability disaster recovery (HADR) operational state of the database. The state between primary and standby servers can be one of the following: Connected, Congested or Disconnected. A change from Connected to another state might generate an Attention alert.

**Identifier**

db.hadr\_op\_status

**Health monitor level**

Database

**Category**

High availability disaster recovery

**Type** State-based

**Unit** Not applicable

*db.hadr\_delay - HADR Log Delay:*

This health indicator tracks the current average delay (in minutes) between the data changes on the primary database and the replication of those changes on the standby database. With a large delay value, data loss can occur when failing over to the standby database after a failure on the primary database. A large delay value can also mean longer downtime when takeover is required, because the primary database is ahead of the standby database.

**Identifier**

db.hadr\_delay

**Health monitor level**

Database

**Category** High availability disaster recovery  
**Type** Upper-bounded threshold-based  
**Unit** Minutes

**Logging health indicators:**

*db.log\_util - Log Utilization:*

This indicator tracks the total amount of active log space used in bytes in the database.

**Identifier**  
db.log\_util

**Health monitor level**  
Database

**Category**  
Logging

**Type** Upper-bounded threshold-based  
**Unit** Percentage

Log utilization is measured as the percentage of space consumed, where a high percentage may generate an alert.

The indicator is calculated using the formula:

$(db.total\_log\_used / (db.total\_log\_used + db.total\_log\_available)) * 100$

The values for the log-related database configuration parameters, shown in the additional information, display the current allocations for logs. The additional information also includes the application id for the application which has the oldest active transaction. This application can be forced to free up log space.

*db.log\_fs\_util - Log File System Utilization:*

Log File System Utilization tracks the fullness of the file system on which the transaction logs reside.

**Identifier**  
db.log\_fs\_util

**Health monitor level**  
Database

**Category**  
Logging

**Type** Upper-bounded threshold-based  
**Unit** Percentage

The DB2 database system may not be able to create a new log file if there is no room on the file system.

Log utilization is measured as the percentage of space consumed. If the amount of free space in the file system is minimal (that is, there is a high percentage for utilization), an alert may be generated.

The indicator is calculated using the formula:  $(fs.log\_fs\_used / fs.log\_fs\_total)*100$  where fs is the file system on which the log resides.

The values for the log-related database configuration parameters, shown in the additional information, display the current allocations for logs. The additional details also shows if user exit is enabled.

If Block on Log Disk Full, shown in the additional details, is set to yes and utilization is at 100%, you should resolve any alerts as soon as possible to limit the impact to applications which cannot commit transactions until the log file is successfully created.

### Application concurrency health indicators:

*db.deadlock\_rate* - Deadlock Rate:

Deadlock rate tracks the rate at which deadlocks are occurring in the database and the degree to which applications are experiencing contention problems.

#### Identifier

db.deadlock\_rate

#### Health monitor level

Database

#### Category

Application Concurrency

**Type** Upper-bounded threshold-based

**Unit** Deadlocks per hour

Deadlocks may be caused by the following situations:

- Lock escalations are occurring for the database
- An application may be locking tables explicitly when system-generated row locks may be sufficient
- An application may be using an inappropriate isolation level when binding
- Catalog tables are locked for repeatable read
- Applications are getting the same locks in different orders, resulting in deadlock.

The indicator is calculated using the formula:

$(db.deadlocks_t - db.deadlocks_{t-1})$

where  $t$  is the current snapshot and  $t-1$  is the last snapshot, taken 60 minutes before the current snapshot.

The higher the rate of deadlocks, the greater the degree of contention which may generate an alert.

*db.locklist\_util* - Lock List Utilization:

This indicator tracks the amount of lock list memory that is being used.

#### Identifier

db.locklist\_util

#### Health monitor level

Database



**Category**

Application Concurrency

**Type** Upper-bounded threshold-based**Unit** Percentage

There is one lock list per database and it contains the locks held by all applications concurrently connected to the database. There is a set limit on lock list memory. Once the limit is reached, performance degrades because of the following situations:

- Lock escalation converts row locks to table locks, thereby reducing concurrency on shared objects in the database.
- More deadlocks between applications can occur since applications are waiting for a limited number of table locks. As a result, transactions are rolled back.

An error is returned to the application when the maximum number of lock requests has reached the limit set for the database.

The indicator is calculated using the formula:

$$(db.lock\_list\_in\_use / (locklist * 4096)) * 100$$

Utilization is measured as a percentage of memory consumed, where a high percentage represents an unhealthy condition.

Consider using the self-tuning memory feature to have lock memory resources automatically allocated as required by the current workload. If you have the self-tuning memory feature enabled for the lock memory area, you should configure this health indicator to disable threshold checking.

*db.lock\_escal\_rate* - Lock Escalation Rate:

This indicator tracks the rate at which locks have been escalated from row locks to a table lock thereby impacting transaction concurrency.

**Identifier**

db.lock\_escal\_rate

**Health monitor level**

Database

**Category**

Application Concurrency

**Type** Upper-bounded threshold-based**Unit** Lock escalations per hour

A lock is escalated when the total number of locks held by an application reaches the maximum amount of lock list space available to the application, or the lock list space consumed by all applications is approaching the total lock list space. The amount of lock list space available is determined by the *maxlocks* and *locklist* database configuration parameters.

When an application reaches the maximum number of locks allowed and there are no more locks to escalate, the application uses the space in the lock list allocated for other applications. There is one lock list per database and it contains the locks held by all applications concurrently connected to the database. When the entire lock list is full, an error occurs.

The indicator is calculated using the formula:

$$(db.lock\_escals_t - db.lock\_escals_{t-1})$$

where 't' is the current snapshot and 't-1' is the last snapshot, taken 60 minutes before the current snapshot.

The higher the rate of deadlocks, the greater the degree of contention which may generate an alert.

Consider using the self-tuning memory feature to have lock memory resources automatically allocated as required by the current workload. If you have the self-tuning memory feature enabled for the lock memory area, you should configure this health indicator to disable threshold checking.

*db.apps\_waiting\_locks - Percentage of Applications Waiting on Locks:*

This indicator measures the percentage of all currently executing applications that are waiting on locks.

**Identifier**

db.apps\_waiting\_locks

**Health monitor level**

Database

**Category**

Application Concurrency

**Type** Upper-bounded threshold-based

**Unit** Percentage

A high percentage can indicate that applications are experiencing concurrency problems which can negatively affect performance.

The indicator is calculated using the formula:

$$(db.locks\_waiting / db.appls\_cur\_cons) * 100$$

**Package cache, catalog cache, and workspace health indicators:**

*db.catcache\_hitratio - Catalog Cache Hit Ratio:*

The hit ratio is a percentage indicating how well the catalog cache is helping to avoid actual accesses to the catalog on disk. A high ratio indicates it is successful in avoiding actual disk I/O accesses.

**Identifier**

db.catcache\_hitratio

**Health monitor level**

Database

**Category**

Package and Catalog Caches, and Workspaces

**Type** Lower-bounded threshold-based

**Unit** Percentage

The indicator is calculated using the formula:

$$(1 - (db.cat\_cache\_inserts/db.cat\_cache\_lookups)) * 100$$

*db.pkgcache\_hitratio - Package Cache Hit Ratio:*

The hit ratio is a percentage indicating how well the package cache is helping to avoid reloading packages and sections for static SQL from the system catalogs as well as helping to avoid recompiling dynamic SQL statements. A high ratio indicates it is successful in avoiding these activities.

**Identifier**

db.pkgcache\_hitratio

**Health monitor level**

Database

**Category**

Package and Catalog Caches, and Workspaces

**Type** Lower-bounded threshold-based

**Unit** Percentage

The indicator is calculated using the formula:

$(1 - (\text{db.pkg\_cache\_inserts} / \text{db.pkg\_cache\_lookups})) * 100$

Consider using the self-tuning memory feature to have package cache memory resources automatically allocated as required by the current workload. If you have the self-tuning memory feature enabled for the package cache memory area, you should configure this health indicator to disable threshold checking.

*db.shrworkspace\_hitratio - Shared Workspace Hit Ratio:*

The hit ratio is a percentage indicating how well the shared SQL workspace is helping to avoid having to initialize sections for SQL statements that are about to be executed. A high ratio indicates it is successful in avoiding this action.

**Identifier**

db.shrworkspace\_hitratio

**Health monitor level**

Database

**Category**

Package and Catalog Caches, and Workspaces

**Type** Lower-bounded threshold-based

**Unit** Percentage

The indicator is calculated using the formula:

$(1 - (\text{db.shr\_workspace\_section\_inserts} / \text{db.shr\_workspace\_section\_lookups})) * 100$

**Memory health indicators:**

*db2.mon\_heap\_util - Monitor Heap Utilization:*

This indicator tracks the consumption of the monitor heap memory, based on the memory pool with the ID SQLM\_HEAP\_MONITOR.

**Identifier**

db2.mon\_heap\_util

**Health monitor level**

Instance

**Category**

Memory

**Type** Upper-bounded threshold-based**Unit** Percentage

The utilization is calculated using the formula:

$$(db2.pool\_cur\_size / db2.pool\_max\_size) * 100$$

for the Memory Pool Identifier SQLM\_HEAP\_MONITOR.

Once this percentage reaches the maximum, 100%, monitor operations may fail.

*db.db\_heap\_util - Database Heap Utilization:*

This indicator tracks the consumption of the monitor heap memory, based on the memory pool with the ID SQLM\_HEAP\_DATABASE.

**Identifier**

db.db\_heap\_util

**Health monitor level**

Database

**Category**

Memory

**Type** Upper-bounded threshold-based**Unit** Percentage

The utilization is calculated using the formula

$$(db.pool\_cur\_size / db.pool\_max\_size) * 100$$

for the Memory Pool Identifier SQLM\_HEAP\_DATABASE.

Once this percentage reaches the maximum, 100%, queries and operations may fail because there is no heap available.

**Federated health indicators:**

*db.fed\_nicknames\_op\_status - Nickname Status:*

This health indicator checks all of the nicknames defined in a federated database to determine if there are any invalid nicknames. A nickname may be invalid if the data source object was dropped or changed, or if the user mapping is incorrect.

**Identifier**

db.fed\_nicknames\_op\_status

**Health monitor level**

Database

**Category**

Federated

**Type** Collection state-based**Unit** Not applicable

An attention alert might be generated if any nicknames defined in the federated database are invalid. Refer to the collection details of this health indicator for the list of objects that need attention.

The FEDERATED database manager parameter must be set to YES for this health indicator to check nicknames status.

*db.fed\_servers\_op\_status - Data Source Server Status:*

This health indicator checks all of the data source servers defined in a federated database to determine if any are unavailable. A data source server might be unavailable if the data source server was stopped, no longer exists, or was incorrectly configured.

**Identifier**

db.fed\_servers\_op\_status

**Health monitor level**

Database

**Category**

Federated

**Type** Collection state-based

**Unit** Not applicable

An attention alert might be generated if any nicknames defined in the federated database are not valid. Refer to the collection details of this health indicator for the list of objects that need attention.

The FEDERATED database manager parameter must be set to YES for this health indicator to check data source server status.

**Health monitor interfaces**

The following table lists the health monitor interfaces for APIs:

**Note:** These APIs have been deprecated and might be removed in a future release because the health monitor has been deprecated in Version 9.7.

*Table 63. Health monitor interfaces: APIs*

<b>Monitoring task</b>	<b>API</b>
Capturing a health snapshot	db2GetSnapshot - Get Snapshot with snapshot class SQLM_CLASS_HEALTH
Capturing a health snapshot with the full list of collection objects	db2GetSnapshot - Get Snapshot with snapshot class SQLM_CLASS_HEALTH and SQLM_HMON_OPT_COLL_FULL for agent_id
Capturing a health snapshot with formula, additional information, and history	db2GetSnapshot - Get Snapshot with snapshot class SQLM_CLASS_HEALTH_WITH_DETAIL
Capturing a health snapshot with formula, additional information, history, and the full list of collection objects	db2GetSnapshot - Get Snapshot with snapshot class SQLM_CLASS_HEALTH_WITH_DETAIL and SQLM_HMON_OPT_COLL_FULL for agent_id
Converting the self-describing data stream	db2ConvMonStream - Convert Monitor stream
Estimating the size of a health snapshot	db2GetSnapshotSize - Estimate Size Required for db2GetSnapshot Output Buffer

The following table lists the health monitor interfaces for CLP commands:

**Note:** These commands have been deprecated and might be removed in a future release because the health monitor has been deprecated in Version 9.7.

*Table 64. Health monitor interfaces: CLP commands*

Monitoring task	CLP command
Capturing a health snapshot	GET HEALTH SNAPSHOT Command
Capturing a health snapshot with formula, additional information, and history	GET HEALTH SNAPSHOT WITH DETAILS Command

The following table lists the health monitor interfaces for SQL functions:

**Note:** These SQL functions have been deprecated and might be removed in a future release because the health monitor has been deprecated in Version 9.7.

*Table 65. Health monitor interfaces: SQL functions*

Monitoring task	SQL Function
Database manager level health information snapshot	HEALTH_DBM_INFO
Database manager level health indicator snapshot	HEALTH_DBM_HI
Database manager level health indicator history snapshot	HEALTH_DBM_HI_HIS
Database level health information snapshot	HEALTH_DB_INFO
Database level health indicator snapshot	HEALTH_DB_HI
Database level health indicator history snapshot	HEALTH_DB_HI_HIS
Database level health indicator collection snapshot	HEALTH_DB_HIC
Database level health indicator collection history snapshot	HEALTH_DB_HIC_HIS
Table space level health information snapshot	HEALTH_TBS_INFO
Table space level health indicator snapshot	HEALTH_TBS_HI
Table space level health indicator history snapshot	HEALTH_TBS_HI_HIS
Table space container level health information snapshot	HEALTH_CONT_INFO
Table space container level health indicator snapshot	HEALTH_CONT_HI
Table space container level health indicator history snapshot	HEALTH_CONT_HI_HIS

### Health monitor SQL table functions:

The following table lists all of the snapshot table functions. Each table function corresponds to a health snapshot request type.

**Note:** These table functions have been deprecated and might be removed in a future release because the health monitor has been deprecated in Version 9.7.

*Table 66. Snapshot monitor SQL table functions*

Monitor level	SQL table function	Information returned
Database manager	HEALTH_DBM_INFO	Basic information about the health snapshot from the database manager level
Database manager	HEALTH_DBM_HI	Health indicator information from the database manager level

Table 66. Snapshot monitor SQL table functions (continued)

Monitor level	SQL table function	Information returned
Database manager	HEALTH_DBM_HI_HIS	Health indicator history information from the database manager level
Database	HEALTH_DB_INFO	Basic information about the health snapshot from a database
Database	HEALTH_DB_HI	Health indicator information from a database
Database	HEALTH_DB_HI_HIS	Health indicator history information from a database
Database	HEALTH_DB_HIC	Collection information for collection health indicators for a database
Database	HEALTH_DB_HIC_HIS	Collection history information for collection health indicators for a database
Table space	HEALTH_TBS_INFO	Basic information about the health snapshot for the table spaces for a database
Table space	HEALTH_TBS_HI	Health indicator information about the table spaces for a database
Table space	HEALTH_TBS_HI_HIS	Health indicator history information about the table spaces for a database
Table space	HEALTH_CONT_INFO	Basic information about the health snapshot for the containers for a database
Table space	HEALTH_CONT_HI	Health indicator information about the containers for a database
Table space	HEALTH_CONT_HI_HIS	Health indicator history information about the containers for a database

#### Health monitor CLP commands:

The following table lists all the supported snapshot request types.

Table 67. Snapshot monitor CLP commands

Monitor level	CLP command	Information returned
Database manager	get health snapshot for dbm	Database manager level information.
Database	get health snapshot for all databases	Database level information. Information is returned only if the database is activated.
Database	get health snapshot for database on <i>database-alias</i>	Database level information. Information is returned only if the database is activated.
Database	get health snapshot for all on <i>database-alias</i>	Database, table space, and table space container information. Information is returned only if the database is activated.
Table space	get snapshot for tablespaces on <i>database-alias</i>	Table space level information for each table space that has been accessed by an application connected to the database. Also includes health information for each table space container within the table space.

#### Health monitor API request types:

The following table lists all the supported snapshot request types.

Table 68. Snapshot Monitor API Request Types

Monitor level	API request type	Information returned
Database manager	SQLMA_DB2	Database manager level information.
Database	SQLMA_DBASE_ALL	Database level information. Information is returned only if the database is activated.
Database	SQLMA_DBASE	Database level information. Information is returned only if the database is activated.
Table space	SQLMA_DBASE_TABLESPACES	Table space level information for each table space that has been accessed by an application connected to the database. Also includes health information for each table space container within the table space.

## Working with the Memory Visualizer

The Memory Visualizer helps database administrators to monitor the memory-related performance of an instance and all of its databases. You can view a live, visual display of the memory utilization of memory components organized in a hierarchical tree.

**Important:** The Memory Visualizer has been deprecated in Version 9.7 and might be removed in a future release. For more information, see the “Control Center tools and DB2 administration server (DAS) have been deprecated” topic in the *What’s New for DB2 Version 9.7* book.

To view memory performance and usage plots and to update the configuration parameters in the Memory Visualizer, you must have SYSADM authority.

You can use the Memory Visualizer to troubleshoot performance problems. You can change the configuration parameter settings for a memory component and assess the effect that the changes have. Configuration parameters affect memory usage in DB2 because memory is allocated as it is required. If you set the value of a configuration parameter above or below its acceptable range, an error message will display. Changing the configuration parameter takes immediate effect within the Memory Visualizer, and the new value is integrated in the next refresh cycle.

- To view memory performance using the Memory Visualizer:
  1. Open the Memory Visualizer from the Windows **Start** menu, by clicking **Programs** → **IBM DB2** → **Monitoring** → **Tools** → **Memory Visualizer**. The Memory Visualizer instance selection window opens. Select an instance from the **Instance name** field and click **OK**.
  2. Expand the instance object tree until you display the databases and their associated memory components in the hierarchical tree. Values for the memory pools display in the Memory Visualizer window.
  3. To display a plotted graph of a memory component, use one of the following methods:
    - Select a component in the hierarchical tree and click the **Show Plot** check box in the Memory Visualizer window.
    - Right-click the selected memory component to display a pop-up menu and select **Show Plot**.



- Select a component in the hierarchical tree and select the **Show Plot** option from the Selected menu on the tool bar. The plotted data for each memory component appears in the Memory Usage Plot.
- To view data from another memory component, select it from the hierarchical tree and click the **Show Plot** check box. The plotted data for the component appears in the Memory Usage Plot along with other components.

The graph displays data collected for memory components over time. Each component is represented by a color and shape which also displays in the **Plot Legend** field in the Memory Visualizer window. The shape is repeated at intervals. A label identifies the component in the graph plot.

The time that the performance data was captured is displayed below the graph. You can change the time interval for the graph.

**Note:** When a new memory component is added to the plot, it does not replace the memory components that were previously added.

Horizontal and vertical scroll bars offer different views of the plotted data.

- Use the horizontal scroll bar, located at the bottom of the graph, to view historical data of the memory component over a selected time period. Point to and drag the slider bar along the base of the graph.
- Use the vertical scroll bar, located at the right of the graph, to view the memory utilization of the selected component. Point to and drag the slider to change the view.

When the memory utilization reaches a new high, the maximum value of the vertical scroll bar is updated to reflect the new value. You can set the minimum value of the vertical scroll bar to a value other than 0 to view a different range of pool utilization values.

- You can load data from a Memory Visualizer data file into a new Memory Visualizer window. This data can be used to compare the performance of an instance and all of its databases against historical data. To load data from a Memory Visualizer data file, select **Open** from the Memory Visualizer menu, and then from the Open Dialog select a data file with extension \*.mdf.
- Use the **Time Unit** field to change the time interval on the Memory Usage Plot window. The default time interval for the graph data is minutes. You can select intervals of minutes, hours, or days. When selected, a new time interval displays in the horizontal range of the graph and changes the incremental movement of the horizontal scroll bar.
- To remove the plotted graph of a memory component from the Memory Usage Plot, either select a component in the hierarchical tree and clear the **Show Plot** check box in the Memory Visualizer window, or right-click the selected memory component to display a pop-up menu and deselect **Show Plot**. The plotted data for the component is removed from the Memory Usage Plot window. The colored shape, which represented the component, no longer displays in the **Plot Legend** field in the Memory Visualizer window.
- To help you to track and create a history of memory performance, you can save memory performance data, including plotted graphs, while the Memory Visualizer is running. To save memory performance data, select **Save** or **Save As** from the Memory Visualizer menu, and then select a location for the file and a filename with extension .mdf.
- To change the configuration parameter settings for a memory component:
  1. Expand the memory pool that you want so that you can see its configuration parameters listed in the hierarchical tree.

2. Click a component to select it and click the number in the **Parameter Value** column. A text box displays the current value for the component. Type a new number in the text box and press **Enter**. The new value displays next to the original value in the **Parameter Value** column until the configuration parameter is updated possibly in the next refresh cycle. You can also right-click the value in the **Parameter Value** column for the selected component to display the pop-up menu. Click outside of the column to complete the change. The new value for the memory component displays next to the original value in the **Parameter Value** column. If you select to view a graph of memory performance, you will see the new value in the graph plot view. While this change takes place immediately in the Memory Visualizer, there is a delay in updating the change you made to the configuration parameter within DB2. You can reset the value of the configuration parameter using the **Reset to Default** option in the pop-up menu.

## Memory Visualizer overview

Use the Memory Visualizer to monitor the memory-related performance of an instance and all of its databases.

**Important:** The Memory Visualizer has been deprecated in Version 9.7 and might be removed in a future release. For more information, see the “Control Center tools and DB2 administration server (DAS) have been deprecated” topic in the *What’s New for DB2 Version 9.7* book.

Open the Memory Visualizer and select a memory component or multiple components in the hierarchical tree to display values for the amount of memory allocated to the component and the current memory usage in the Memory Visualizer window. The Memory Visualizer window displays two views of data: a tree view and a historical view. A series of columns show percentage threshold values for upper and lower alarms and warnings. The columns also display real time memory utilization.

**Note:** The Memory Visualizer is available to provide memory performance data for instances that are Version 8.1 and later.

The following list categorizes some of the key tasks that you can do with the Memory Visualizer:

- View or hide data in various columns on the memory utilization of selected components for a DB2 instance and its databases.
- View a graph of memory performance data.
- Change settings for individual memory components by updating configuration parameters.
- Load performance data from a file into a Memory Visualizer window.
- Save the memory performance data.

The Memory Visualizer interface has the following elements that help you monitor the memory-related performance of an instance and all of its databases.

### The Memory Visualizer window

The columns in the Memory Visualizer window display values for the performance of memory components. The following information is shown:

#### Plot Legend

The checked memory components or configuration parameters

shown in the Memory Usage Plot. A specific shape that occurs at regular intervals in the plotted graph identifies each component or parameter.

**Utilization**

The size of the memory that is allocated to, and utilized by, the database object. Includes a graphical bar showing the utilization and configured allocation. The length of the bar is fixed and the filled portion indicates utilization as a percentage.

**Parameter Value**

The current value of a configuration parameter.

**Upper Alarm (%) Threshold**

The threshold value that generates an upper alarm. The default value is 98%.

**Upper Warning (%) Threshold**

The threshold value that generates an upper warning. The default value is 90%.

**Lower Alarm (%) Threshold**

The threshold value that generates a lower alarm. The default value is 2%.

**Lower Warning (%) Threshold**

The threshold value that generates a lower warning. The default value is 10%.

**Graphical Usage Bars**

The graphical usage bars in the Memory Visualizer window are visual cues of memory utilization. The bars can assist you in determining how much memory is being used by selected memory components and the potential effect that the usage can have on the system. The Memory Visualizer also displays a percentage value that corresponds to the usage. These two indicators can help you to determine whether you need to change the configuration parameter setting for the component or take another appropriate action.

**Memory Components**

The Database Manager uses different types of memory on a system, namely Database manager shared memory, Database global memory, Application global memory, Agent /Application shared memory, and Agent private memory. These types of memory are the high level memory components that the Memory Visualizer uses in its expanding hierarchical tree organization.

Underlying each high-level memory component are other components that determine how the memory is allocated and deallocated. For example, when the database manager starts, a database is activated, an application connects to a database, or when an agent is assigned to work for an application, memory is allocated and deallocated. The Memory Visualizer uses these leaf-level memory components to display how memory is allocated and used in a DB2 instance.

**Hierarchical Tree Organization**

The Memory Visualizer uses a hierarchical tree organization to help you to display and browse the memory components in DB2. The hierarchical tree

allows you to expand and view information on individual memory components through columns, graphical displays, and graphs.

The tree view comprises four major types of memory items:

**DB2 Instance**

The instance that is currently running on the system

**Databases**

The databases defined on the instance





**High-level memory components**

Logical groupings for leaf-level memory components. These groups are: Database manager shared memory, Database global memory, Agent private memory, Agent /Application shared memory

**Leaf-level memory components**

The memory components that display in the Memory Visualizer window such as buffer pools, sort heaps, database heap, and lock list.

Icons in the tree view represent each memory tree item:

- Instance: 
- Database: 
- High-level memory groupings: 
- Leaf-level memory components: 

If the memory utilization for a tree items exceeds a threshold value, a colored indicator overlays the icon. The yellow color indicates a warning condition. The red color indicates an alarm condition.

The historical view displays data for memory components selected in the tree view. The data includes values for memory allocated and utilized, plotted graphs, as well as changes made to the configuration parameters while the Memory Visualizer is running. The data is saved for a specific period within the Memory Visualizer. You can save memory performance data to a Memory Visualizer data file for tracking, comparing with other data, or troubleshooting.

**The Memory Usage Graph**

The memory usage graph displays plotted data for selected memory components in the Memory Usage Plot. Each component in the graph is identified by a specific color, which also displays in the Plot Legend column in the Memory Visualizer window. The graph also displays changes made to the configuration parameters settings. The original value of the configuration parameter and the new value setting appear in the graph, in addition to the time that the change was requested. They become part of the history view that you can use in assessing memory performance.

For further details, see “Working with the Memory Visualizer” on page 185.

---

## Activity Monitor overview

Use the Activity Monitor to monitor application performance and concurrency, resource consumption, and SQL statement usage of a database or database partition.

**Important:** The Activity Monitor has been deprecated in Version 9.7 and might be removed in a future release. For more information, see the “Control Center tools and DB2 administration server (DAS) have been deprecated” topic in the *What’s New for DB2 Version 9.7* book.

The Activity Monitor provides a set of predefined reports based on a specific subset of monitor data. These reports allow you to focus monitoring on application performance, application concurrency, resource consumption, and SQL statement use. The Activity Monitor also provides recommendations for most reports. These recommendations can assist you to diagnose the cause of database performance problems, and to tune queries for optimal utilization of database resources.

Figure 7 on page 191 describes the process for using the Activity monitor to solve a problem.

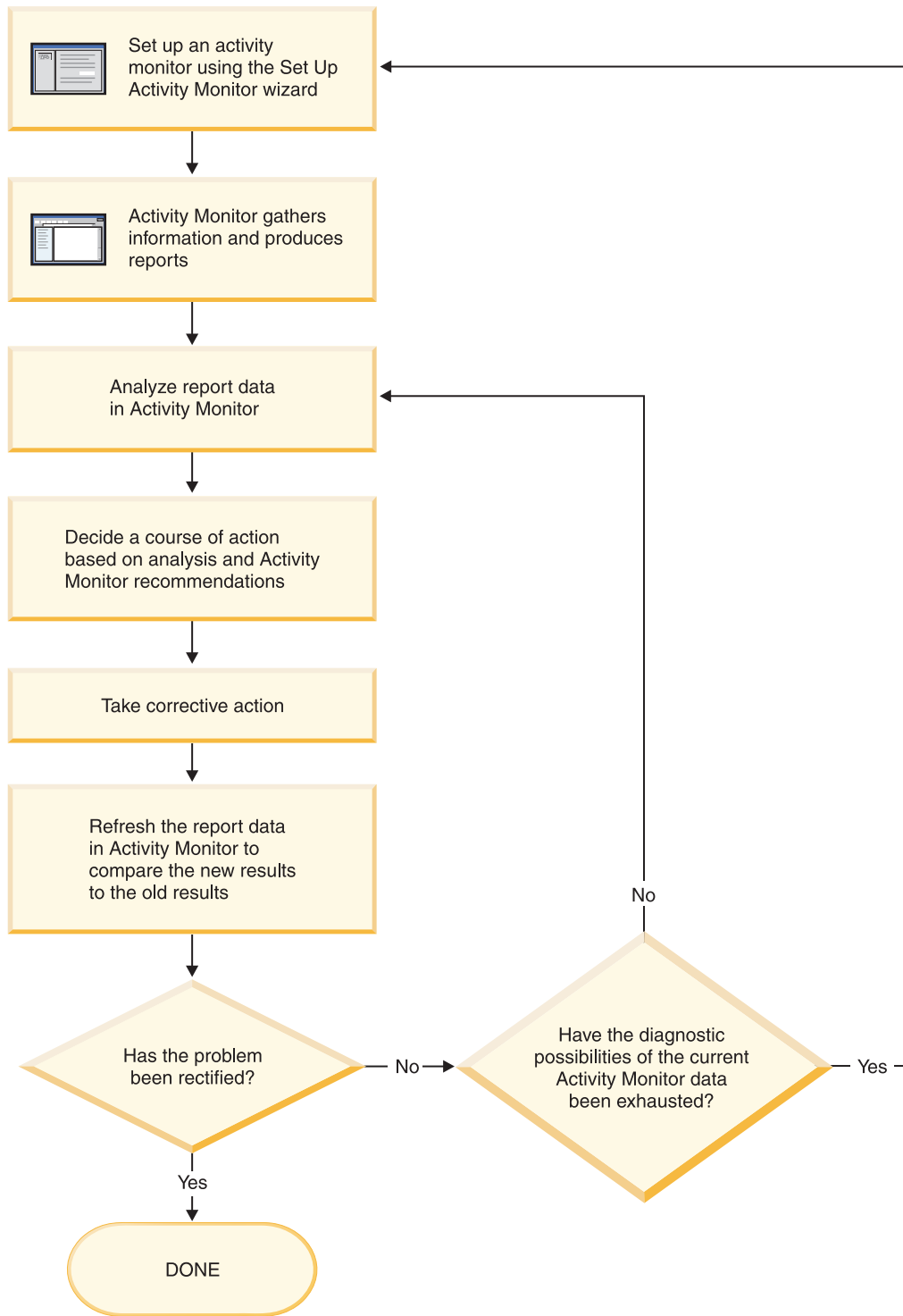


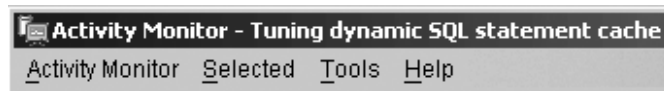
Figure 7. Activity Monitor overview

Table 69. Tasks that you can perform from the Activity Monitor

Tasks from the Activity Monitor	Aspects of tasks	Invocation
Transactions	View transactions running on a selected application.	Select one or more applications in the <b>Report data</b> pane. Right-click and select <b>Show Latest Transactions</b> . The Application Transactions window opens.
Statements	View SQL statements running on a selected application.	Select one or more applications in the <b>Report data</b> pane. Right-click and select <b>Show Latest Statements</b> . The Application Statements window opens.
	View the text of SQL statements running on a selected application.	From the Application Statements window, right-click on a statement in the <b>Report data</b> pane. Select <b>Show Statement Text</b>
Application Lock Chains	View locks and lock-waiting situations that currently affect a selected application.	Select an application in the <b>Report data</b> pane. Right-click and select <b>Show Lock Chains</b> . The Application Lock Chains window opens.
	View information about a selected application for which you are viewing lock information.	From the Application Lock Chains window, right-click an application, and select <b>About</b> .
	View information about the locks held and the locks waited on by a selected application in your database.	From the Application Lock Chains window, right-click an application, and select <b>Show Lock Details</b> .
View report data and recommendations	View information to help you interpret report data.	From an Activity Monitor window, an Application Statements window, or an Application Transactions window, use the <b>Report</b> arrow to select the report and click the <b>Report Details</b> push button. View the <b>Details</b> page.
	View recommendations provided by Activity Monitor	From an Activity Monitor window, an Application Statements window, or an Application Transactions window, use the <b>Report</b> arrow to select the report and click the <b>Report Details</b> push button. View the <b>Recommendations</b> page.

The Activity Monitor interface has several elements that help you organize and interpret the monitor data that is collected:

## Menu bar



Use the menu bar to work with objects in the Activity Monitor, open other administration centers and tools, and access online help.

## Activity Monitor toolbar



Use the toolbar icons to open DB2 tools and view DB2 information.

## Report data pane

Application Handle (agent ID)	Application Name	Authorization ID	Application ID	Total CPU Time	User CPU Time
18	acmerpt.exe	EDWARDL	*LOCAL.DB2.00...	180259	10014
20	db2cc.exe	DB2ADMIN	*LOCAL.DB2.00...	30042	10014
22	acmefin.exe	FREDS	*LOCAL.DB2.00...	20028	20028
21	db2evm.exe	DB2ADMIN	*LOCAL.DB2.00...	20028	10014
27	acmeacct.exe	ALICET	*LOCAL.DB2.00...	10015	10015

Use the **Report data** pane to display and to work with the report data that is available to you within the Activity Monitor. The **Report data** pane displays the items that make up the contents of the report that is selected in the **Report** field.

The **Report data** pane also provides access to other Activity Monitor windows. From the Activity Monitor, you can drill down from the applications you are monitoring to the individual transactions or to the individual SQL statements that these applications are running.

## Report data pane toolbar



Use the toolbar located below the **Report data** pane to tailor the view of objects and information in the **Report data** pane to suit your needs.

## Monitoring scenarios

### Scenario: Identifying costly applications using snapshot administrative views

Recent increases in the workload on the ShopMart database have started hindering overall database performance. Jessie, the ShopMart DBA, is trying to identify the larger resource consumers in the daily workload using the following administrative views:

#### APPLICATION\_PERFORMANCE

This view helps Jessie identify applications that might be performing large table scans:

```
connect to shopmart;  
select AGENT_ID, ROWS_SELECTED, ROWS_READ from APPLICATION_PERFORMANCE;
```

The value of ROWS\_SELECTED shows her how many rows are returned to an application and the value of ROWS\_READ shows her how many rows are accessed from the base tables. If the selectivity is low, the application might be performing a table scan that could be avoided with



the creation of an index. Jessie uses this view to identify potentially troublesome queries, and then she can investigate further by looking at the SQL to see if there are any ways to reduce the number of rows that are read in the execution of the query.

### **LONG\_RUNNING\_SQL**

Jessie uses the LONG\_RUNNING\_SQL administrative view to identify the longest running queries that are currently being executed:

```
connect to shopmart;
select ELAPSED_TIME_MIN, APPL_STATUS, AGENT_ID
  from long_running_sql order by ELAPSED_TIME_MIN desc
  fetch first 5 rows only;
```

Using this view, she can determine the length of time these queries have been running, and the status of these queries. If a query has been executing for a long time and is waiting on a lock, she can use the LOCKWAITS or LOCK\_HELD administrative views querying on a specific agent id to investigate further. The LONG\_RUNNING\_SQL view can also tell her the statement that is being executed, allowing her to identify potentially problematic SQL.

### **QUERY\_PREP\_COST**

Jessie uses the QUERY\_PREP\_COST to troubleshoot queries that have been identified as problematic. This view can tell her how frequent a query is run as well as the average execution time for each of these queries:

```
connect to shopmart;
select NUM_EXECUTIONS, AVERAGE_EXECUTION_TIME_S, PREP_TIME_PERCENT
  from QUERY_PREP_COST order by NUM_EXECUTIONS desc;
```

The value of PREP\_TIME\_PERCENT tells Jessie what percentage of the queries execution time is spent preparing the query. If the time it takes to compile and optimize a query is almost as long as it takes for the query to execute, Jessie might want to advise the owner of the query to change the optimization class used for the query. Lowering the optimization class might make the query complete optimization more rapidly and therefore return a result sooner. However, if a query takes a significant amount of time to prepare but is executed thousands of times (without being prepared again) then changing the optimization class might not benefit query performance.

### **TOP\_DYNAMIC\_SQL**

Jessie uses the TOP\_DYNAMIC\_SQL view to identify the most frequently executed, longest-running and most sort-intensive dynamic SQL statements. Having this information will allow Jessie to focus her SQL tuning efforts on the queries that represent some of the biggest resource consumers

To identify the most frequently run dynamic SQL statements, Jessie issues the following:

```
connect to shopmart;
select * from TOP_DYNAMIC_SQL order by NUM_EXECUTIONS desc
  fetch first 5 rows only;
```

This returns all of the details regarding the execution time, number of sorts performed, and the statement text for the five most frequent dynamic SQL statements.

To identify the dynamic SQL statements with the longest execution times, Jessie examines the queries with the top five values for `AVERAGE_EXECUTION_TIME_S`:

```
connect to shopmart;  
select * from TOP_DYNAMIC_SQL  
order by AVERAGE_EXECUTION_TIME_S desc fetch first 5 rows only;
```

To look at the details of the most sort-intensive dynamic SQL statements, Jessie issues the following:

```
connect to shopmart;  
select STMT_SORTS, SORTS_PER_EXECUTION, substr(STMT_TEXT,1,60) as STMT_TEXT  
from TOP_DYNAMIC_SQL order by STMT_SORTS desc  
fetch first 5 rows only;
```

### **Scenario: Monitoring buffer pool efficiency using administrative views**

John, a DBA, suspects that poor application performance in the SALES database is a result of buffer pools that function inefficiently. To investigate, he takes a look at the buffer pool hit ratio using the `BP_HITRATIO` administrative view:

```
connect to SALES;  
select BPNAME, TOTAL_HIT_RATIO from BP_HIT_RATIO;
```

John sees that the hit ratio for one of the buffer pools is very low, which means that too many pages are being read from disk instead of being read from the buffer pool.

He then decides to use the `BP_READ_IO` administrative view to see whether the prefetchers require tuning:

```
connect to SALES;  
select BPNAME, PERCENT_SYNC_READS, UNUSED_ASYNC_READS_PERCENT from BP_READ_IO;
```

The value for `PERCENT_SYNC_READS` tells him the percentage of pages read synchronously without prefetching. A high number indicates that a high percentage of data is being read directly from disk, and might indicate that more prefetchers are required. The value of `UNUSED_ASYNC_READS_PERCENT` tells him the percentage of pages read asynchronously from disk, but never accessed by a query. This might indicate that the prefetchers are overly aggressive in reading in data pages, resulting in unnecessary I/O.

Since both the values for `PERCENT_SYNC_READS` and `UNUSED_ASYNC_READS_PERCENT` seem within the acceptable range, John uses the `BP_WRITE_IO` administrative view to investigate how well the page cleaners are working to clear space for incoming data pages:

```
connect to SALES;  
select BPNAME, PERCENT_WRITES_ASYNC from BP_WRITE_IO;
```

The value of `PERCENT_WRITES_ASYNC` tells John what percentage of physical write requests that were performed asynchronously. If this number is high, it might mean that the page cleaners are working well to clear space in the buffer pool ahead of incoming requests for new data pages. If this number is low, then a higher number of physical writes are being performed by database agents while an application waits for data a data page to be read into the buffer pool.

John sees that the value of `PERCENT_WRITES_ASYNC` is very low at 25 percent, so he decides to configure more page cleaners for the SALES database to increase the rate of asynchronous writes. After increasing the number of page cleaners, he can use the buffer pool administrative views again to see the effects of his tuning.

## Setting up an activity monitor

To monitor application performance and concurrency, resource consumption, and SQL statement usage of a database or database partition, you can set up an activity monitor. The Activity Monitor provides a set of predefined reports based on a specific subset of monitor data. It can also provide recommendations to assist you in diagnosing the cause of database performance problems, and to tune queries so that your use of database resources is optimized.

**Important:** The Activity Monitor has been deprecated in Version 9.7 and might be removed in a future release. For more information, see the “Control Center tools and DB2 administration server (DAS) have been deprecated” topic in the *What’s New for DB2 Version 9.7* book.

To use the Activity Monitor:

- Your server must have DB2 UDB Version 8.2 or later
- You must have SQLADM or DBADM authority

Open the Set Up Activity Monitor wizard.

- From the Control Center, expand the object tree until you find the instance or the database for which you want to set up an activity monitor. Right-click the object and select Set Up Activity Monitor from the pop-up menu.
- From the command line, type the following command: `db2am`.

Detailed information is provided through the contextual help facility within the Control Center.

## Progress monitoring of the rollback process

If you obtain an application snapshot while a transaction is rolling back, you will see rollback monitor elements in the output. This information can be used to monitor the progress of the rollback operation.

The information provided in the application snapshot includes the start time of the rollback, the total work to be done, and completed work. The work metric is bytes.

The following is an example of output from the `GET SNAPSHOT FOR ALL APPLICATIONS` command:

### Application Snapshot

```
Application handle      = 6
Application status     = Rollback Active
  Start Time           = 02/20/2004 12:49:27.713720
  Completed Work       = 1024000 bytes
  Total Work           = 4084000 bytes
```

### Application Snapshot

```
Application handle      = 10
Application status     = Rollback to Savepoint
  Start Time           = 02/20/2004 12:49:32.832410
  Completed Work       = 102400 bytes
  Total Work           = 2048000 bytes
```

The value in the application status monitor element implies which type of rollback event is occurring:

**Rollback Active**

This is a unit of work rollback: an explicit (user invoked) or implicit (forced) rollback of the entire transaction.

**Savepoint rollback**

This is a partial rollback to a statement or application level savepoint. Nested savepoints are considered a single unit, using the outermost savepoint.

Completed Work units shows the relative position in the log stream that has been rolled back. Updates to Completed Work are made after every log record is processed. Updates are not performed evenly because log records vary in size.

Total Work units is an estimate based on the range of log records in the log stream that need to be rolled back for the transaction or savepoint. It does not indicate the exact number of log record bytes that need to be processed.

## Using snapshot monitor data to monitor the reorganization of a partitioned table

The following information describes some of the most useful methods of monitoring the global status of a table reorganization.

There is no separate data group indicating the overall table reorganization status for a partitioned table. A partitioned table uses a data organization scheme in which table data is divided across multiple storage objects, called data partitions or ranges, according to values in one or more table partitioning key columns of the table. However, you can deduce the global status of a table reorganization from the values of elements in the individual data partition data group being reorganized. The following information describes some of the most useful methods of monitoring the global status of a table reorganization.

**Determining the number of data partitions being reorganized**

You can determine the total number of data partitions being reorganized on a table by counting the number of monitor data blocks for table data that have the same table name and schema name. This value indicates the number of data partitions on which reorganization has started. Examples 1 and 2 indicate that three data partitions are being reorganized.

**Identifying the data partition being reorganized**

You can deduce the current data partition being reorganized from the phase start time (`reorg_phase_start`). During the SORT/BUILD/REPLACE phase, the monitor data corresponding to the data partition that is being reorganized shows the most recent phase start time. During the INDEX\_RECREATE phase, the phase start time is the same for all the data partitions. In Examples 1 and 2, the INDEX\_RECREATE phase is indicated, so the start time is the same for all the data partitions.

**Identifying an index rebuild requirement**

You can determine if an index rebuild is required by obtaining the value of the maximum reorganize phase element (`reorg_max_phase`), corresponding to any one of the data partitions being reorganized. If `reorg_max_phase` has a value of 3 or 4, then an Index Rebuild is required. Examples 1 and 2 report a `reorg_max_phase` value of 3, indicating an index rebuild is required.

The following sample output is from a three-node server that contains a table with three data partitions:

```
CREATE TABLE sales (c1 INT, c2 INT, c3 INT)
PARTITION BY RANGE (c1)
(PART P1 STARTING FROM (1) ENDING AT (10) IN parttbs,
PART P2 STARTING FROM (11) ENDING AT (20) IN parttbs,
PART P3 STARTING FROM (21) ENDING AT (30) IN parttbs)
DISTRIBUTE BY (c2)
```

Statement executed:

```
REORG TABLE sales ALLOW NO ACCESS ON ALL DBPARTITIONNUMS
```

*Example 1:*

```
GET SNAPSHOT FOR TABLES ON DPARTDB GLOBAL
```

The output is modified to include table information for the relevant table only.

#### Table Snapshot

```
First database connect timestamp = 06/28/2005 13:46:43.061690
Last reset timestamp            = 06/28/2005 13:46:47.440046
Snapshot timestamp              = 06/28/2005 13:46:50.964033
Database name                    = DPARTDB
Database path                    = /work/sales/NODE0000/SQL00001/
Input database alias             = DPARTDB
Number of accessed tables        = 5
```

#### Table List

```
Table Schema      = NEWTON
Table Name        = SALES
Table Type        = User
Data Partition Id = 0
Data Object Pages = 3
Rows Read         = 12
Rows Written      = 1
Overflows         = 0
Page Reorgs      = 0
Table Reorg Information:
Node number       = 0
Reorg Type        =
  Reclaiming
  Table Reorg
  Allow No Access
  Recluster Via Table Scan
  Reorg Data Only
Reorg Index       = 0
Reorg Tablespace = 3
Long Temp space ID = 3
Start Time        = 06/28/2005 13:46:49.816883
Reorg Phase       = 3 - Index Recreate
Max Phase         = 3
Phase Start Time  = 06/28/2005 13:46:50.362918
Status            = Completed
Current Counter   = 0
Max Counter       = 0
Completion        = 0
End Time          = 06/28/2005 13:46:50.821244
```

#### Table Reorg Information:

```
Node number       = 1
Reorg Type        =
  Reclaiming
  Table Reorg
  Allow No Access
```

```

Recluster Via Table Scan
Reorg Data Only
Reorg Index      = 0
Reorg Tablespace = 3
Long Temp space ID = 3
Start Time       = 06/28/2005 13:46:49.822701
Reorg Phase      = 3 - Index Recreate
Max Phase        = 3
Phase Start Time = 06/28/2005 13:46:50.420741
Status           = Completed
Current Counter  = 0
Max Counter      = 0
Completion       = 0
End Time         = 06/28/2005 13:46:50.899543

```

```

Table Reorg Information:
Node number      = 2
Reorg Type       =
  Reclaiming
  Table Reorg
  Allow No Access
  Recluster Via Table Scan
  Reorg Data Only
Reorg Index      = 0
Reorg Tablespace = 3
Long Temp space ID = 3
Start Time       = 06/28/2005 13:46:49.814813
Reorg Phase      = 3 - Index Recreate
Max Phase        = 3
Phase Start Time = 06/28/2005 13:46:50.344277
Status           = Completed
Current Counter  = 0
Max Counter      = 0
Completion       = 0
End Time         = 06/28/2005 13:46:50.803619

```

```

Table Schema     = NEWTON
Table Name       = SALES
Table Type       = User
Data Partition Id = 1
Data Object Pages = 3
Rows Read        = 8
Rows Written     = 1
Overflows        = 0
Page Reorgs     = 0
Table Reorg Information:
Node number      = 0
Reorg Type       =
  Reclaiming
  Table Reorg
  Allow No Access
  Recluster Via Table Scan
  Reorg Data Only
Reorg Index      = 0
Reorg Tablespace = 3
Long Temp space ID = 3
Start Time       = 06/28/2005 13:46:50.014617
Reorg Phase      = 3 - Index Recreate
Max Phase        = 3
Phase Start Time = 06/28/2005 13:46:50.362918
Status           = Completed
Current Counter  = 0
Max Counter      = 0
Completion       = 0
End Time         = 06/28/2005 13:46:50.821244

```

```

Table Reorg Information:
Node number      = 1
Reorg Type      =
  Reclaiming
  Table Reorg
  Allow No Access
  Recluster Via Table Scan
  Reorg Data Only
Reorg Index      = 0
Reorg Tablespace = 3
Long Temp space ID = 3
Start Time       = 06/28/2005 13:46:50.026278
Reorg Phase      = 3 - Index Recreate
Max Phase        = 3
Phase Start Time = 06/28/2005 13:46:50.420741
Status           = Completed
Current Counter  = 0
Max Counter      = 0
Completion       = 0
End Time         = 06/28/2005 13:46:50.899543

```

```

Table Reorg Information:
Node number      = 2
Reorg Type      =
  Reclaiming
  Table Reorg
  Allow No Access
  Recluster Via Table Scan
  Reorg Data Only
Reorg Index      = 0
Reorg Tablespace = 3
Long Temp space ID = 3
Start Time       = 06/28/2005 13:46:50.006392
Reorg Phase      = 3 - Index Recreate
Max Phase        = 3
Phase Start Time = 06/28/2005 13:46:50.344277
Status           = Completed
Current Counter  = 0
Max Counter      = 0
Completion       = 0
End Time         = 06/28/2005 13:46:50.803619

```

```

Table Schema      = NEWTON
Table Name        = SALES
Table Type        = User
Data Partition Id = 2
Data Object Pages = 3
Rows Read         = 4
Rows Written      = 1
Overflows         = 0
Page Reorgs      = 0
Table Reorg Information:
Node number      = 0
Reorg Type      =
  Reclaiming
  Table Reorg
  Allow No Access
  Recluster Via Table Scan
  Reorg Data Only
Reorg Index      = 0
Reorg Tablespace = 3
Long Temp space ID = 3
Start Time       = 06/28/2005 13:46:50.199971
Reorg Phase      = 3 - Index Recreate
Max Phase        = 3

```

```

Phase Start Time = 06/28/2005 13:46:50.362918
Status           = Completed
Current Counter  = 0
Max Counter     = 0
Completion       = 0
End Time        = 06/28/2005 13:46:50.821244

```

Table Reorg Information:

```

Node number      = 1
Reorg Type      =
  Reclaiming
  Table Reorg
  Allow No Access
  Recluster Via Table Scan
  Reorg Data Only
Reorg Index     = 0
Reorg Tablespace = 3
Long Temp space ID = 3
Start Time      = 06/28/2005 13:46:50.223742
Reorg Phase     = 3 - Index Recreate
Max Phase       = 3
Phase Start Time = 06/28/2005 13:46:50.420741
Status          = Completed
Current Counter  = 0
Max Counter     = 0
Completion       = 0
End Time        = 06/28/2005 13:46:50.899543

```

Table Reorg Information:

```

Node number      = 2
Reorg Type      =
  Reclaiming
  Table Reorg
  Allow No Access
  Recluster Via Table Scan
  Reorg Data Only
Reorg Index     = 0
Reorg Tablespace = 3
Long Temp space ID = 3
Start Time      = 06/28/2005 13:46:50.179922
Reorg Phase     = 3 - Index Recreate
Max Phase       = 3
Phase Start Time = 06/28/2005 13:46:50.344277
Status          = Completed
Current Counter  = 0
Max Counter     = 0
Completion       = 0
End Time        = 06/28/2005 13:46:50.803619

```

*Example 2:*

**GET SNAPSHOT FOR TABLES ON DPARTDB AT DBPARTITIONNUM 2**

The output is modified to include table information for the relevant table only.

Table Snapshot

```

First database connect timestamp = 06/28/2005 13:46:43.617833
Last reset timestamp             =
Snapshot timestamp              = 06/28/2005 13:46:51.016787
Database name                   = DPARTDB
Database path                   = /work/sales/NODE0000/SQL00001/
Input database alias            = DPARTDB
Number of accessed tables       = 3

```

Table List

```

Table Schema = NEWTON

```



```

Table Name          = SALES
Table Type          = User
Data Partition Id   = 0
Data Object Pages   = 1
Rows Read           = 0
Rows Written        = 0
Overflows           = 0
Page Reorgs         = 0
Table Reorg Information:
  Node number       = 2
  Reorg Type        =
    Reclaiming
    Table Reorg
    Allow No Access
    Recluster Via Table Scan
    Reorg Data Only
  Reorg Index       = 0
  Reorg Tablespace = 3
Long Temp space ID = 3
Start Time          = 06/28/2005 13:46:49.814813
Reorg Phase         = 3 - Index Recreate
Max Phase           = 3
Phase Start Time    = 06/28/2005 13:46:50.344277
Status              = Completed
Current Counter     = 0
Max Counter         = 0
Completion          = 0
End Time            = 06/28/2005 13:46:50.803619

```

```

Table Schema        = NEWTON
Table Name          = SALES
Table Type          = User
Data Partition Id   = 1
Data Object Pages   = 1
Rows Read           = 0
Rows Written        = 0
Overflows           = 0
Page Reorgs         = 0
Table Reorg Information:
  Node number       = 2
  Reorg Type        =
    Reclaiming
    Table Reorg
    Allow No Access
    Recluster Via Table Scan
    Reorg Data Only
  Reorg Index       = 0
  Reorg Tablespace = 3
Long Temp space ID = 3
Start Time          = 06/28/2005 13:46:50.006392
Reorg Phase         = 3 - Index Recreate
Max Phase           = 3
Phase Start Time    = 06/28/2005 13:46:50.344277
Status              = Completed
Current Counter     = 0
Max Counter         = 0
Completion          = 0
End Time            = 06/28/2005 13:46:50.803619

```

```

Table Schema        = NEWTON
Table Name          = SALES
Table Type          = User
Data Partition Id   = 2
Data Object Pages   = 1
Rows Read           = 4

```

```

Rows Written          = 1
Overflows             = 0
Page Reorgs          = 0
Table Reorg Information:
  Node number         = 2
  Reorg Type          =
    Reclaiming
    Table Reorg
    Allow No Access
    Recluster Via Table Scan
    Reorg Data Only
  Reorg Index         = 0
  Reorg Tablespace    = 3
Long Temp space ID    = 3
Start Time            = 06/28/2005 13:46:50.179922
Reorg Phase           = 3 - Index Recreate
Max Phase             = 3
Phase Start Time      = 06/28/2005 13:46:50.344277
Status                = Completed
Current Counter       = 0
Max Counter           = 0
Completion            = 0
End Time              = 06/28/2005 13:46:50.803619

```

Example 3:

```
SELECT * FROM SYSIBMADM.SNAPLOCK WHERE tabname = 'SALES';
```

The output is modified to include a subset of table information for the relevant table only.

...	TBSP_NAME	TABNAME	LOCK_OBJECT_TYPE	LOCK_MODE	LOCK_STATUS	...
...	PARTTBS	SALES	ROW_LOCK	X	GRNT	...
...	-	SALES	TABLE_LOCK	IX	GRNT	...
...	PARTTBS	SALES	TABLE_PART_LOCK	IX	GRNT	...
...	PARTTBS	SALES	ROW_LOCK	X	GRNT	...
...	-	SALES	TABLE_LOCK	IX	GRNT	...
...	PARTTBS	SALES	TABLE_PART_LOCK	IX	GRNT	...
...	PARTTBS	SALES	ROW_LOCK	X	GRNT	...
...	-	SALES	TABLE_LOCK	IX	GRNT	...
...	PARTTBS	SALES	TABLE_PART_LOCK	IX	GRNT	...

9 record(s) selected.

Output from this query (continued).

...	LOCK_ESCALATION	LOCK_ATTRIBUTES	DATA_PARTITION_ID	DBPARTITIONNUM
...	0	INSERT	2	2
...	0	NONE	-	2
...	0	NONE	2	2
...	0	INSERT	0	0
...	0	NONE	-	0
...	0	NONE	0	0
...	0	INSERT	1	1
...	0	NONE	-	1
...	0	NONE	1	1

Example 4:

```
SELECT * FROM SYSIBMADM.SNAPTAB WHERE tabname = 'SALES';
```

The output is modified to include a subset of table information for the relevant table only.

```

... TABSCHEMA TABNAME TAB_FILE_ID TAB_TYPE DATA_OBJECT_PAGES ROWS_WRITTEN ...
... -----
... NEWTON SALES 2 USER_TABLE 1 1 ...
... NEWTON SALES 4 USER_TABLE 1 1 ...
... NEWTON SALES 3 USER_TABLE 1 1 ...

```

3 record(s) selected.

Output from this query (continued).

```

... OVERFLOW_ACCESSES PAGE_REORGS DBPARTITIONNUM TBSP_ID DATA_PARTITION_ID
... -----
... 0 0 0 3 0
... 0 0 2 3 2
... 0 0 1 3 1

```

*Example 5:*

```
SELECT * FROM SYSIBMADM.SNAPTAB_REORG WHERE tabname = 'SALES';;
```

The output is modified to include a subset of table information for the relevant table only.

```

REORG_PHASE REORG_MAX_PHASE REORG_TYPE ...
-----
INDEX_RECREATE 3 RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY ...
INDEX_RECREATE 3 RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY ...
INDEX_RECREATE 3 RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY ...
INDEX_RECREATE 3 RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY ...
INDEX_RECREATE 3 RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY ...
INDEX_RECREATE 3 RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY ...
INDEX_RECREATE 3 RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY ...
INDEX_RECREATE 3 RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY ...
INDEX_RECREATE 3 RECLAIM+OFFLINE+ALLOW_NONE+TABLESCAN+DATAONLY ...

```

9 record(s) selected.

Output from this query (continued).

```

... REORG_STATUS REORG_TBSPC_ID DBPARTITIONNUM DATA_PARTITION_ID
... -----
... COMPLETED 3 2 0
... COMPLETED 3 2 1
... COMPLETED 3 2 2
... COMPLETED 3 1 0
... COMPLETED 3 1 1
... COMPLETED 3 1 2
... COMPLETED 3 0 0
... COMPLETED 3 0 1
... COMPLETED 3 0 2

```

*Example 6:* The Table Reorg Information includes information about reclaiming extents as part of a reorganization operation. The example that follows shows the relevant output.

```
db2 -v "get snapshot for tables on wsdb"
```

```

Table Reorg Information:
  Reorg Type          =
    Reclaim Extents
    Allow Write Access
  Reorg Index         = 0
  Reorg Tablespace    = 0
  Start Time          = 10/22/2008 15:49:35.477532
  Reorg Phase         = 12 - Release
  Max Phase           = 3

```

**Note:** Any snapshot requests from a monitor version before SQLM\_DBMON\_VERSION9\_7 will not return any Reclaim Reorg status to the requesting client.

## Inactive statement tracking for DEADLOCK WITH DETAILS HISTORY event monitors

When running a deadlock event monitor that tracks all statements (and optionally data values) it becomes possible for the system monitor heap to be exhausted by a single application that includes a very high number of statements in a unit of work. It is also possible for the monitor heap to become exhausted if there are a large number of applications executing concurrently.

To mitigate the amount of space that is consumed, inactive statements are written out to the event monitor by an application when the number of inactive statements for that application reaches a certain threshold. After being written to the event monitor, the memory consumed by these inactive statements will be released. Also, if at any time an application is unable to acquire memory from the system monitor heap, that application writes out all of its current inactive statements to the event monitor before trying to acquire the memory again. If the second attempt fails, a message is logged, and the statement history list is truncated for the UOW the application is processing.

The default limit on the number of inactive statements to be kept by any one application is 250. This default value can be overridden using the registry variable DB2\_MAX\_INACT\_STMTS to specify a different value. Users may want to choose a different value for the limit to increase or reduce the amount of system monitor heap used for inactive statement information.

Whenever inactive statements are written out to the event monitor, a message appears in the db2diag log file indicating that this has occurred. Whenever the limit for inactive statements has been exceeded, a message appears in the db2diag log file indicating that this has occurred.

Because an application can now record its statement history entries outside of the context of a deadlock (when one of the thresholds mentioned above are reached), we need a mechanism to associate these entries to the list of statements recorded at the time of a deadlock for analysis. To do this, users can look for statement history entries where:

- `deadlock_id= 0`
- `participant_no = 0`
- `invocation_id= invocation id of the deadlock`
- `application_id= application identifier of the application that participated in the deadlock`

In the case of a write to table event monitor, the number of `evmon_activates` also needs to be checked.

### Notes:

- For SQL statements compiled using the REOPT ALWAYS bind option, there will be no reopt compilation or statement execution data values provided in the deadlock event information.
- At coordinator nodes, when inactive statements are written to the event monitor due to the conditions described in the previous section, the sequence value of all records written will be changed to reflect the current unit of work in process.

This is done to help reconcile this data with any data generated later by a deadlock in the same unit of work since all the relevant data can be gathered by searching for the sequence number and application ID information for those records with `deadlock_id` of 0. This change does mean that the unit of work information is not available for statements started in a preceding unit of work but still active in the current work, as the sequence number will be overwritten by the current unit of work identifier. This behavior does not occur at remote nodes (that is, the original unit of work information is not overwritten) and so care must be taken when trying to reconcile deadlock event records with any records written out prior to the deadlock as the sequence numbers might differ if there are active cursors with hold from previous units of work involved.

---

## Introduction to Windows Management Instrumentation (WMI)

There is an industry initiative that establishes management infrastructure standards and provides a way to combine information from various hardware and software management systems. This initiative is called Web-Based Enterprise Management (WBEM). WBEM is based on the Common Information Model (CIM) schema, which is an industry standard driven by the Desktop Management Task Force (DMTF).

Microsoft® Windows Management Instrumentation (WMI) is an implementation of the WBEM initiative for supported Windows platforms. WMI is useful in a Windows enterprise network where it reduces the maintenance and cost of managing enterprise network components. WMI provides:

- A consistent model of Windows operation, configuration, and status.
- A COM API to allow access to management information.
- The ability to operate with other Windows management services.
- A flexible and extensible architecture allowing vendors a means of writing other WMI providers to support new devices, applications, and other enhancements.
- The WMI Query Language (WQL) to create detailed queries of the information.
- An API for management application developers to write Visual Basic or Windows Scripting Host (WSH) scripts.

The WMI architecture has two parts:

1. A management infrastructure that includes the CIM Object Manager (CIMOM) and a central storage area for management data called the CIMOM object repository. CIMOM allows applications to have a uniform way to access management data.
2. WMI providers. WMI providers are the intermediaries between CIMOM and managed objects. Using WMI APIs, WMI providers supply CIMOM with data from managed objects, handle requests on behalf of management applications, and generate event notifications.

Windows Management Instrumentation (WMI) providers are standard COM or DCOM servers that function as mediators between managed objects and the CIM Object Manager (CIMOM). If the CIMOM receives a request from a management application for data that is not available from the CIMOM object repository, or for events, the CIMOM forwards the request to the WMI providers. WMI providers supply data, and event notifications, for managed objects that are specific to their particular domain.

## DB2 database system integration with Windows Management Instrumentation

The snapshot monitors can be accessed by Windows Management Instrumentation (WMI) by means of the DB2 performance counters and using the built-in PerfMon provider.

The DB2 profile registry variables can be accessed by WMI by using the built-in Registry provider.

The WMI Software Development Kit (WMI SDK) includes several built-in providers:

- PerfMon provider
- Registry event provider
- Registry provider
- Windows event log provider
- Win32 provider
- WDM provider

The DB2 errors that are in the Event Logs can be accessed by WMI by using the built-in Windows Event Log provider.

DB2 database system has a DB2 WMI Administration provider, and sample WMI script files, to access the following managed objects:

1. Instances of the database server including those instances that are distributed. The following operations can be done:
  - Enumerate instances
  - Configure database manager parameters
  - Start/stop/query the status of the DB2 server service
  - Setup or establish communication
2. Databases. The following operations can be done:
  - Enumerate databases
  - Configure database parameters
  - Create/drop databases
  - Backup/restore/roll forward databases

You will need to register the DB2 WMI provider with the system before running WMI applications. Registration is done by entering the following commands:

- `mofcomp %DB2PATH%\bin\db2wmi.mof`  
This command loads the definition of the DB2 WMI schema into the system.
- `regsvr %DB2PATH%\bin\db2wmi.dll`  
This command registers the DB2 WMI provider COM DLL with Windows.

In both commands, %DB2PATH% is the path where DB2 is installed. Also, db2wmi.mof is the .MOF file that contains the DB2 WMI schema definition.

There are several benefits to integrating with the WMI infrastructure:

1. You are able to easily write scripts to manage DB2 servers in a Windows-based environment using the WMI provided tool. Sample Visual Basic (VBS) scripts are provided to carry out simple tasks such as listing instances, creating and

dropping databases, and updating configuration parameters. The sample scripts are included in the DB2 Application Development for Windows product.

2. You can create powerful management applications that perform many tasks using WMI. The tasks could include:
  - Displaying system information
  - Monitoring DB2 performance
  - Monitoring DB2 system resource consumption

By monitoring both system events and DB2 events through this type of management application, you can manage the database better.

3. You can use existing COM and Visual Basic programming knowledge and skills. By providing a COM or Visual Basic interface, your programmers can save time when developing enterprise management applications.

## Windows performance monitor introduction

When working with DB2 database manager for Windows, there are tools that can be used to monitor performance:

- **DB2 Performance Expert**

DB2 Performance Expert for Multiplatforms, Version 1.1 consolidates, reports, analyzes and recommends self-managing and resource tuning changes based on DB2 database performance-related information.

- **DB2 Health Center**

The functions of the Health Center provide you with different methods to work with performance-related information. These functions somewhat replace the performance monitor capability of the Control Center.

**Important:** The Health Center has been deprecated in Version 9.7 and might be removed in a future release. For more information, see the “Control Center tools and DB2 administration server (DAS) have been deprecated” topic in the *What’s New for DB2 Version 9.7* book.

- **Windows Performance Monitor**

The Windows Performance Monitor enables you to monitor both database and system performance, retrieving information from any of the performance data providers registered with the system. Windows also provides performance information data on all aspects of computer operation including:

- CPU usage
- Memory utilization
- Disk activity
- Network activity

## Registering DB2 with the Windows performance monitor

The setup program automatically registers DB2 with the Windows Performance Monitor for you.

To make DB2 database and DB2® Connect™ performance information accessible to the Windows Performance Monitor, you must register the DLL for the DB2 for Windows Performance Counters. This also enables any other Windows application using the Win32 performance APIs to get performance data. To install and register the DB2 Performance Counters DLL (DB2Perf.DLL) with the Windows Performance Monitor, type:

```
db2perfi -i
```

Registering the DLL also creates a new key in the services option of the registry. One entry gives the name of the DLL, which provides the counter support. Three other entries give names of functions provided within that DLL. These functions include:

**Open** Called when the DLL is first loaded by the system in a process.

**Collect**

Called to request performance information from the DLL.

**Close** Called when the DLL is unloaded.

## Enabling remote access to DB2 performance information

If your DB2 for Windows workstation is networked to other Windows computers, you can use the feature described in this section.

In order to see Windows performance objects from another DB2 for Windows computer, you must register an administrator username and password with the DB2 database manager. (The default Windows Performance Monitor username, SYSTEM, is a DB2 database reserved word and cannot be used.) To register the name, type:

```
db2perfr -r username password
```

**Note:** The username used must conform to the DB2 database naming rules.

The username and password data is held in a key in the registry, with security that allows access only by administrators and the SYSTEM account. The data is encoded to prevent security concerns about storing an administrator password in the registry.

**Note:**

1. Once a username and password combination has been registered with the DB2 database system, even local instances of the Performance Monitor will explicitly log on using that username and password. This means that if the username information registered with DB2 database system does not match, local sessions of the Performance Monitor will not show DB2 database performance information.
2. The username and password combination must be maintained to match the username and password values stored in the Windows Security database. If the username or password is changed in the Windows Security database, the username and password combination used for remote performance monitoring must be reset.
3. To deregister, type:

```
db2perfr -u <username> <password>
```

## Displaying DB2 database and DB2 Connect performance values

To display DB2 database and DB2 Connect performance values using the Performance Monitor, simply choose the performance counters whose values you want displayed from the **Add to** box. This box displays a list of performance objects providing performance data. Select an object to see a list of the counters it supplies.



A performance object can also have multiple instances. For example, the LogicalDisk object provides counters such as “% Disk Read Time” and “Disk Bytes/sec”; it also has an instance for each logical drive in the computer, including “C:” and “D:”.

## Windows performance objects

Windows provides the following performance objects:

- **DB2 Database Manager**

This object provides general information for a single Windows instance. The DB2 database instance being monitored appears as the object instance.

For practical and performance reasons, you can only get performance information from one DB2 database instance at a time. The DB2 database instance that the Performance Monitor shows is governed by the db2instance registry variable in the Performance Monitor process. If you have multiple DB2 database instances running simultaneously and want to see performance information from more than one, you must start a separate session of the Performance Monitor, with db2instance set to the relevant value for each DB2 database instance to be monitored.

If you are running a partitioned database environment, you can only get performance information from one database partition server at a time. By default, the performance information for the default database partition (that is, the database partition that has logical port 0) is displayed. To see performance information of another database partition, you must start a separate session of the Performance Monitor with the DB2NODE environment variable set to the database partition number of the database partition to be monitored.

- **DB2 Databases**

This object provides information for a particular database. Information is available for each currently active database.

- **DB2 Applications**

This object provides information for a particular DB2 database application. Information is available for each currently active DB2 database application.

- **DB2 DCS Databases**

This object provides information for a particular DCS database. Information is available for each currently active database.

- **DB2 DCS Applications**

This object provides information for a particular DB2 DCS application. Information is available for each currently active DB2 DCS application.

Which of these objects will be listed by the Windows Performance Monitor depends on what is installed on your Windows computer and what applications are active. For example, if the DB2 database manager is installed has been started, the DB2 Database Manager object will be listed. If there are also some DB2 databases and applications currently active on that computer, the DB2 Databases and DB2 Applications objects will be listed as well. If you are using your Windows system as a DB2 Connect gateway and there are some DCS databases and applications currently active, the DB2 DCS Databases and DB2 DCS Applications objects will be listed.

## Accessing remote DB2 database performance information

Enabling remote access to DB2 Performance Information was discussed earlier. In the **Add to** box, select another computer to monitor. This brings up a list of all the available performance objects on that computer.

In order to be able to monitor DB2 Performance object on a remote computer, the level of the DB2 database or DB2 Connect code installed on that computer must be Version 6 or higher.

## Resetting DB2 performance values

When an application calls the DB2 monitor APIs, the information returned is normally the cumulative values since the DB2 database server was started. However, often it is useful to:

- Reset performance values
- Run a test
- Reset the values again
- Re-run the test.

To reset database performance values, use the `db2perf` program. Type:

```
db2perf
```

By default, this resets performance values for all active DB2 databases. However, you can also specify a list of databases to reset. You can also use the `-d` option to specify that performance values for DCS databases should be reset. For example:

```
db2perf
db2perf dbalias1 dbalias2 ... dbaliasn

db2perf -d
db2perf -d dbalias1 dbalias2 ... dbaliasn
```

The first example resets performance values for all active DB2 databases. The next example resets values for specific DB2 databases. The third example resets performance values for all active DB2 DCS databases. The last example resets values for specific DB2 DCS databases.

The `db2perf` program resets the values for ALL programs currently accessing database performance information for the relevant DB2 database server instance (that is, the one held in `DB2INSTANCE` in the session in which you run `db2perf`).

Invoking `db2perf` also resets the values seen by anyone remotely accessing DB2 database performance information when the `db2perf` command is executed.

**Note:** There is a DB2 database API, `sqlmrset`, that allows an application to reset the values it sees locally, not globally, for particular databases.

---

## Indoubt Transaction Manager overview

Use the Indoubt Transaction Manager window to work with indoubt transactions. The window lists all indoubt transactions for a selected database and one or more selected partitions.

**Important:** The Indoubt Transaction Manager has been deprecated in Version 9.7 and might be removed in a future release. For more information, see the “Control Center tools and DB2 administration server (DAS) have been deprecated” topic in the *What’s New for DB2 Version 9.7* book.

An indoubt transaction is a global transaction that was left in an indoubt state. DB2 provides heuristic actions that database administrators can perform on indoubt transactions when the resource owner, such as the database administrator, cannot wait for the Transaction Manager to perform the resync action. This condition may occur if, for example, the communication line is broken, and an indoubt transaction is tying up needed resources such as locks on tables and indexes, log space, and storage used by the transaction.

While it is preferable for the Transaction Manager to initiate the re-sync action, there may be times when you may have to perform the heuristic actions on the indoubt transactions. In these cases, use the heuristic actions with caution and only as a last resort and follow these guidelines.

- The *gtrid* portion of the transaction ID is the global transaction ID that is identical to that in other resource managers (RM) that participate in the global transaction.
- Use your knowledge of the application and the operating environment to identify the other participating resource managers,
- If the transaction manager is CICS<sup>®</sup>, and the only resource manager is a CICS resource, perform a heuristic rollback.
- If the transaction manager is not CICS, use it to determine the status of the transaction that has the same *gtrid* as the indoubt transaction.
- If, at least, one resource manager has committed or rolled back, perform a heuristic commit or rollback.
- If all the transactions are in the prepared state, perform a heuristic rollback.
- If, at least, one of the resource managers is not available, perform a heuristic rollback.

To open the Indoubt Transaction Manager on Intel platforms, from the **Start** menu, click **Start -> Programs -> IBM DB2 -> Monitoring Tools -> Indoubt Transaction Manager**.

To open the Indoubt Transaction Manager using the command line in UNIX or on Intel, run the following command:

```
db2indbt
```

You can perform the following heuristic actions on indoubt transactions:

- **Forget**  
This permits the resource manager to erase knowledge of a heuristically completed transaction by removing the log records and releasing log pages. A heuristically completed transaction is one that has been committed or rolled back heuristically. You can use the forget action on transactions that are heuristically committed or rolled back for a selected database and one or more selected partitions. To forget an indoubt transaction, select a database and partition and then right-click a transaction with a status of **Committed** or **Rolled back** and select **Forget** from the pop-up menu. A confirmation message displays.
- **Commit**  
This commits an indoubt transaction that is prepared to be committed. If the operation succeeds, the transaction’s state becomes heuristically committed. To

commit an indoubt transaction, select a database and partition and then right-click a transaction with a status of **Indoubt** or **Missing commit acknowledgement** and select **Commit** from the pop-up menu. A confirmation message displays.

- **Rollback**

This rolls back an indoubt transaction that has been prepared. If the operation succeeds, the transaction's state becomes heuristically rolled back. To roll back an indoubt transaction, select a database and partition and then right-click a transaction with a status of **Indoubt** or **Ended** and select **Rollback** from the pop-up menu. A confirmation message displays.

To perform these actions on indoubt transactions you must have SYSADM or DBADM authority.

The columns in the Indoubt Transaction Manager window provide named views that you can use to organize and display indoubt transactions in different ways. The following list describes each of the columns in the interface:

**Status**

The indoubt status of the transaction, namely Committed (c), Ended (e), Indoubt (i), Missing commit acknowledgement (m), and Rolled back (r):

**Committed**

Transactions in this state have been heuristically committed.

**Ended**

Transactions in this state may have timed out.

**Indoubt**

Transactions in this state are waiting to be committed or rolled back.

**Missing commit acknowledgement**

The Transaction Manager is waiting to receive an acknowledgement before committing the transaction.

**Rolled back**

Transactions in this state have been heuristically rolled back

**Timestamp**

The time stamp on the server when the transaction entered the prepared (indoubt) state. The time is the local time to the client.

**Transaction ID**

The XA identifier assigned by the transaction manager to uniquely identify a global transaction.

**Application ID**

The application identifier assigned by the database manager for this transaction.

**Authorization ID**

The user ID of the user who ran the transaction.

**Sequence Number**

The sequence number assigned by the database manager as an extension to the application identifier.

**Partition**

The partition on which the indoubt transaction exists.

**Originator**

Indicates whether the transaction was originated by XA or by DB2 in a partitioned database environment.

**Log Full**

Indicates whether this transaction caused a log full condition.

**Type** The type information that shows the role of the database in each indoubt transaction.

- **TM** indicates the indoubt transaction is using the database as a transaction manager database.
- **RM** indicates the indoubt transaction is using the database as a resource manager. This means that it is one of the databases participating in the transaction, but is not the transaction manager database.

---

## Part 2. Monitor elements



---

## Chapter 7. Monitor elements reported in monitor table functions

DB2 Version 9.7 introduced a number of monitor elements that are reported through new monitor table functions.

These monitor elements provide information about system processing, activities, and data objects, such as tables, table spaces, table space containers, and buffer pools.

- “act\_aborted\_total - Total aborted activities monitor element” on page 302
- “act\_completed\_total - Total completed activities monitor element” on page 303
- “act\_rejected\_total - Total rejected activities monitor element” on page 305
- “activity\_id - Activity ID monitor element” on page 309
- “activity\_state - Activity state monitor element” on page 310
- “activity\_type - Activity type monitor element” on page 311
- “activitytotaltime\_threshold\_id - Activity total time threshold ID monitor element” on page 311
- “activitytotaltime\_threshold\_value - Activity total time threshold value monitor element” on page 312
- “activitytotaltime\_threshold\_violated - Activity total time threshold violated monitor element” on page 312
- “agent\_id - Application handle (agent ID) monitor element” on page 313
- “agent\_wait\_time - Agent wait time monitor element” on page 316
- “agent\_waits\_total - Total agent waits monitor element” on page 318
- “aggsqltemp space\_threshold\_id - Aggregate SQL temporary space threshold ID monitor element” on page 322
- “aggsqltemp space\_threshold\_value - AggSQL temporary space threshold value monitor element” on page 323
- “aggsqltemp space\_threshold\_violated - AggSQL temporary space threshold violated monitor element” on page 323
- “app\_rqsts\_completed\_total - Total application requests completed monitor element” on page 323
- “application\_handle - Application handle monitor element” on page 333
- “audit\_events\_total - Total audit events monitor element” on page 336
- “audit\_file\_write\_wait\_time - Audit file write wait time monitor element” on page 336
- “audit\_file\_writes\_total - Total audit files written monitor element” on page 337
- “audit\_subsystem\_wait\_time - Audit subsystem wait time monitor element” on page 338
- “audit\_subsystem\_waits\_total - Total audit subsystem waits monitor element” on page 340
- “auto\_storage\_hybrid - Hybrid automatic storage table space indicator monitor element” on page 343
- “automatic - Buffer pool automatic monitor element” on page 343
- “block\_ios - Number of block I/O requests monitor element” on page 344



- “boundary\_leaf\_node\_splits - Boundary leaf node splits monitor element” on page 346
- “bp\_name - Buffer pool name monitor element” on page 347
- “client\_acctng - Client accounting string monitor element” on page 353
- “client\_applname - Client application name monitor element” on page 354
- “client\_idle\_wait\_time - Client idle wait time monitor element” on page 355
- “client\_userid - Client user ID monitor element” on page 358
- “client\_wrkstnname - Client workstation name monitor element” on page 359
- “comp\_env\_desc - Compilation environment monitor element” on page 362
- “concurrentdbcoordactivities\_db\_threshold\_id - Concurrent database coordinator activities database threshold ID monitor element” on page 365
- “concurrentdbcoordactivities\_db\_threshold\_queued - Concurrent database coordinator activities database threshold queued monitor element” on page 365
- “concurrentdbcoordactivities\_db\_threshold\_value - Concurrent database coordinator activities database threshold value monitor element” on page 365
- “concurrentdbcoordactivities\_db\_threshold\_violated - Concurrent database coordinator activities database threshold violated monitor element” on page 366
- “concurrentdbcoordactivities\_subclass\_threshold\_id - Concurrent database coordinator activities service subclass threshold ID monitor element” on page 366
- “concurrentdbcoordactivities\_subclass\_threshold\_queued - Concurrent database coordinator activities service subclass threshold queued monitor element” on page 367
- “concurrentdbcoordactivities\_subclass\_threshold\_value - Concurrent database coordinator activities service subclass threshold value monitor element” on page 367
- “concurrentdbcoordactivities\_subclass\_threshold\_violated - Concurrent database coordinator activities service subclass threshold violated monitor element” on page 367
- “concurrentdbcoordactivities\_superclass\_threshold\_id - Concurrent database coordinator activities service superclass threshold ID monitor element” on page 368
- “concurrentdbcoordactivities\_superclass\_threshold\_queued - Concurrent database coordinator activities service superclass threshold queued monitor element” on page 368
- “concurrentdbcoordactivities\_superclass\_threshold\_value - Concurrent database coordinator activities service superclass threshold value monitor element” on page 369
- “concurrentdbcoordactivities\_superclass\_threshold\_violated - Concurrent database coordinator activities service superclass threshold violated monitor element” on page 369
- “concurrentdbcoordactivities\_work\_action\_set\_threshold\_id - Concurrent database coordinator activities work action set threshold ID monitor element” on page 369
- “concurrentdbcoordactivities\_work\_action\_set\_threshold\_queued - Concurrent database coordinator activities work action set threshold queued monitor element” on page 370
- “concurrentdbcoordactivities\_work\_action\_set\_threshold\_value - Concurrent database coordinator activities work action set threshold value monitor element” on page 370

- “concurrentdbcoordactivities\_work\_action\_set\_threshold\_violated - Concurrent database coordinator activities work action set threshold violated monitor element” on page 371
- “container\_accessible - Accessibility of container monitor element” on page 373
- “container\_id - Container identification monitor element” on page 373
- “container\_name - Container name monitor element” on page 374
- “container\_stripe\_set - Container stripe set monitor element” on page 374
- “container\_total\_pages - Total pages in container monitor element” on page 375
- “container\_type - Container type monitor element” on page 375
- “container\_usable\_pages - Usable pages in container monitor element” on page 375
- “coord\_member - Coordinator member monitor element” on page 380
- “cputime\_threshold\_id - CPU time threshold ID monitor element” on page 385
- “cputime\_threshold\_value - CPU time threshold value monitor element” on page 385
- “cputime\_threshold\_violated - CPU time threshold violated monitor element” on page 385
- “cputimeinsc\_threshold\_id - CPU time in service class threshold ID monitor element” on page 386
- “cputimeinsc\_threshold\_value - CPU time in service class threshold value monitor element” on page 386
- “cputimeinsc\_threshold\_violated - CPU time in service class threshold violated monitor element” on page 386
- “current\_extent - Extent currently being moved monitor element” on page 389
- “data\_partition\_id - Data partition identifier monitor element” on page 390
- “db\_storage\_path\_state - Storage path state monitor element” on page 395
- “db\_storage\_path\_with\_dpe - Storage path including database partition expression monitor element” on page 395
- “db\_work\_action\_set\_id - Database work action set ID monitor element” on page 396
- “db\_work\_class\_id - Database work class ID monitor element” on page 396
- “deadlocks - Deadlocks detected monitor element” on page 399
- “del\_keys\_cleaned - Pseudo deleted keys cleaned monitor element” on page 401
- “diaglog\_write\_wait\_time - Diagnostic log file write wait time monitor element” on page 402
- “diaglog\_writes\_total - Total diagnostic log file writes monitor element” on page 403
- “direct\_read\_reqs - Direct read requests monitor element” on page 404
- “direct\_read\_time - Direct read time monitor element” on page 406
- “direct\_reads - Direct reads from database monitor element” on page 407
- “direct\_write\_reqs - Direct write requests monitor element” on page 409
- “direct\_write\_time - Direct write time monitor element” on page 410
- “direct\_writes - Direct writes to database monitor element” on page 412
- “eff\_stmt\_text - Effective statement text monitor element” on page 415
- “effective\_isolation - Effective isolation monitor element” on page 415
- “effective\_lock\_timeout - Effective lock timeout monitor element” on page 416
- “effective\_query\_degree - Effective query degree monitor element” on page 416

- “empty\_pages\_deleted - Empty pages deleted monitor element” on page 417
- “empty\_pages\_reused - Empty pages reused monitor element” on page 417
- “entry\_time - Entry time monitor element” on page 417
- “estimatedsqlcost\_threshold\_id - Estimated SQL cost threshold ID monitor element” on page 418
- “estimatedsqlcost\_threshold\_value - Estimated SQL cost threshold value monitor element” on page 418
- “estimatedsqlcost\_threshold\_violated - Estimated SQL cost threshold violated monitor element” on page 418
- “executable\_id - Executable ID monitor element” on page 420
- “fcm\_message\_rcv\_volume - FCM message received volume monitor element” on page 422
- “fcm\_message\_rcv\_wait\_time - FCM message received wait time monitor element” on page 423
- “fcm\_message\_recvs\_total - Total FCM message receives monitor element” on page 424
- “fcm\_message\_send\_volume - FCM message send volume monitor element” on page 424
- “fcm\_message\_send\_wait\_time - FCM message send wait time monitor element” on page 425
- “fcm\_message\_sends\_total - Total FCM message sends monitor element” on page 426
- “fcm\_rcv\_volume - FCM received volume monitor element” on page 426
- “fcm\_rcv\_wait\_time - FCM received wait time monitor element” on page 427
- “fcm\_recvs\_total - FCM receives total monitor element” on page 429
- “fcm\_send\_volume - FCM send volume monitor element” on page 430
- “fcm\_send\_wait\_time - FCM send wait time monitor element” on page 430
- “fcm\_sends\_total - FCM sends total monitor element” on page 431
- “fcm\_tq\_rcv\_volume - FCM table queue received volume monitor element” on page 433
- “fcm\_tq\_rcv\_wait\_time - FCM table queue received wait time monitor element” on page 433
- “fcm\_tq\_recvs\_total - FCM table queue receives total monitor element” on page 434
- “fcm\_tq\_send\_volume - FCM table queue send volume monitor element” on page 435
- “fcm\_tq\_send\_wait\_time - FCM table queue send wait time monitor element” on page 436
- “fcm\_tq\_sends\_total - FCM table queue send total monitor element” on page 437
- “files\_closed - Database files closed monitor element” on page 438
- “fs\_caching - File system caching monitor element” on page 440
- “fs\_id - Unique file system identification number monitor element” on page 440
- “fs\_total\_size - Total size of a file system monitor element” on page 441
- “fs\_used\_size - Amount of space used on a file system monitor element” on page 442
- “iid - Index identifier monitor element” on page 458
- “include\_col\_updates - Include column updates monitor element” on page 460
- “index\_only\_scans - Index-only scans monitor element” on page 460

- “index\_scans - Index scans monitor element” on page 460
- “index\_tbsp\_id - Index table space ID monitor element” on page 460
- “insert\_timestamp - Statement insert timestamp monitor element” on page 462
- “int\_node\_splits - Intermediate node splits monitor element” on page 465
- “invocation\_id - Invocation ID monitor element” on page 468
- “ipc\_rcv\_volume - Interprocess communication received volume monitor element” on page 468
- “ipc\_rcv\_wait\_time - Interprocess communication received wait time monitor element” on page 469
- “ipc\_rcvs\_total - Interprocess communication receives total monitor element” on page 469
- “ipc\_send\_volume - Interprocess communication send volume monitor element” on page 470
- “ipc\_send\_wait\_time - Interprocess communication send wait time monitor element” on page 471
- “ipc\_sends\_total - Interprocess communication send total monitor element” on page 472
- “key\_updates - Key updates monitor element” on page 473
- “last\_extent - Last extent moved monitor element” on page 474
- “last\_reference\_time - Last reference time monitor element” on page 475
- “local\_start\_time - Local start time monitor element” on page 477
- “lock\_escals - Number of lock escalations monitor element” on page 480
- “lock\_timeouts - Number of lock timeouts monitor element” on page 488
- “lock\_wait\_time - Time waited on locks monitor element” on page 490
- “lock\_waits - Lock waits monitor element” on page 492
- “log\_buffer\_wait\_time - Log buffer wait time monitor element” on page 495
- “log\_disk\_wait\_time - Log disk wait time monitor element” on page 496
- “log\_disk\_waits\_total - Total log disk waits monitor element” on page 497
- “long\_tbsp\_id - Long table space ID monitor element” on page 501
- “member - Database member monitor element” on page 514
- “nesting\_level - Nesting level monitor element” on page 516
- “nleaf - Number of leaf pages monitor element” on page 517
- “nlevels - Number of index levels monitor element” on page 517
- “nonboundary\_leaf\_node\_splits - Non-boundary leaf node splits monitor element” on page 518
- “num\_executions - Statement executions monitor element” on page 520
- “num\_exec\_with\_metrics - Number of executions with metrics collected monitor element” on page 520
- “num\_extents\_left - Number of extents left to process monitor element” on page 520
- “num\_extents\_moved - Number of extents moved monitor element” on page 521
- “num\_log\_buffer\_full - Number of full log buffers monitor element” on page 522
- “num\_remaps - Number of remaps monitor element” on page 525
- “overflow\_accesses - Accesses to overflowed records monitor element” on page 533
- “overflow\_creates - Overflow creates monitor element” on page 534
- “package\_name - Package name monitor element” on page 534

- “package\_schema - Package schema monitor element” on page 535
- “package\_version\_id - Package version monitor element” on page 535
- “page\_allocations - Page allocations monitor element” on page 536
- “pages\_from\_block\_ios - Total number of pages read by block I/O monitor element” on page 536
- “pages\_from\_vectored\_ios - Total number of pages read by vectored I/O monitor element” on page 537
- “pages\_merged - Pages merged monitor element” on page 538
- “pages\_read - Number of pages read monitor element” on page 538
- “pages\_written - Number of pages written monitor element” on page 538
- “parent\_activity\_id - Parent activity ID monitor element” on page 538
- “parent\_uow\_id - Parent unit of work ID monitor element” on page 539
- “pool\_async\_data\_read\_reqs - Buffer pool asynchronous read requests monitor element” on page 546
- “pool\_async\_data\_reads - Buffer pool asynchronous data reads monitor element” on page 547
- “pool\_async\_data\_writes - Buffer pool asynchronous data writes monitor element” on page 547
- “pool\_async\_index\_read\_reqs - Buffer pool asynchronous index read requests monitor element” on page 548
- “pool\_async\_index\_reads - Buffer pool asynchronous index reads monitor element” on page 549
- “pool\_async\_index\_writes - Buffer pool asynchronous index writes monitor element” on page 550
- “pool\_async\_xda\_read\_reqs - Buffer pool asynchronous XDA read requests monitor element” on page 552
- “pool\_async\_xda\_reads - Buffer pool asynchronous XDA data reads monitor element” on page 553
- “pool\_async\_xda\_writes - Buffer pool asynchronous XDA data writes monitor element” on page 554
- “pool\_data\_l\_reads - Buffer pool data logical reads monitor element” on page 556
- “pool\_data\_p\_reads - Buffer pool data physical reads monitor element” on page 557
- “pool\_data\_writes - Buffer pool data writes monitor element” on page 559
- “pool\_drty\_pg\_steal\_clns - Buffer pool victim page cleaners triggered monitor element” on page 561
- “pool\_drty\_pg\_thrsh\_clns - Buffer pool threshold cleaners triggered monitor element” on page 562
- “pool\_index\_l\_reads - Buffer pool index logical reads monitor element” on page 564
- “pool\_index\_p\_reads - Buffer pool index physical reads monitor element” on page 566
- “pool\_index\_writes - Buffer pool index writes monitor element” on page 567
- “pool\_lsn\_gap\_clns - Buffer pool log space cleaners triggered monitor element” on page 569
- “pool\_no\_victim\_buffer - Buffer pool no victim buffers monitor element” on page 570

- “pool\_read\_time - Total buffer pool physical read time monitor element” on page 571
- “pool\_temp\_data\_l\_reads - Buffer pool temporary data logical reads monitor element” on page 573
- “pool\_temp\_data\_p\_reads - Buffer pool temporary data physical reads monitor element” on page 575
- “pool\_temp\_index\_l\_reads - Buffer pool temporary index logical reads monitor element” on page 577
- “pool\_temp\_index\_p\_reads - Buffer pool temporary index physical reads monitor element” on page 578
- “pool\_temp\_xda\_l\_reads - Buffer pool temporary XDA data logical reads monitor element” on page 580
- “pool\_temp\_xda\_p\_reads - Buffer pool temporary XDA data physical reads monitor element” on page 582
- “pool\_write\_time - Total buffer pool physical write time monitor element” on page 584
- “pool\_xda\_l\_reads - Buffer pool XDA data logical reads monitor element” on page 585
- “pool\_xda\_p\_reads - Buffer pool XDA data physical reads monitor element” on page 588
- “pool\_xda\_writes - Buffer pool XDA data writes monitor element” on page 589
- “post\_shrthreshold\_sorts - Post shared threshold sorts monitor element” on page 591
- “post\_threshold\_sorts - Post threshold sorts monitor element” on page 593
- “prep\_time - Preparation time monitor element” on page 595
- “pseudo\_deletes - Pseudo deletes monitor element” on page 602
- “pseudo\_empty\_pages - Pseudo empty pages monitor element” on page 602
- “qp\_query\_id - Query patroller query ID monitor element” on page 603
- “query\_cost\_estimate - Query cost estimate monitor element” on page 604
- “reclaimable\_space\_enabled - Reclaimable space enabled indicator monitor element” on page 609
- “root\_node\_splits - Root node splits monitor element” on page 621
- “routine\_id - Routine ID monitor element” on page 621
- “rows\_deleted - Rows deleted monitor element” on page 621
- “rows\_inserted - Rows inserted monitor element” on page 622
- “rows\_modified - Rows modified monitor element” on page 623
- “rows\_read - Rows read monitor element” on page 624
- “rows\_returned - Rows returned monitor element” on page 626
- “rows\_updated - Rows updated monitor element” on page 628
- “rqsts\_completed\_total - Total requests completed monitor element” on page 630
- “sc\_work\_action\_set\_id - Service class work action set ID monitor element” on page 630
- “sc\_work\_class\_id - Service class work class ID monitor element” on page 631
- “section\_number - Section number monitor element” on page 633
- “section\_type - Section type indicator monitor element” on page 633
- “service\_class\_id - Service class ID monitor element” on page 639
- “service\_subclass\_name - Service subclass name monitor element” on page 640

- “service\_superclass\_name - Service superclass name monitor element” on page 640
- “sort\_overflows - Sort overflows monitor element” on page 645
- “sqlrowsread\_threshold\_id - SQL rows read threshold ID monitor element” on page 651
- “sqlrowsread\_threshold\_value - SQL rows read threshold value monitor element” on page 651
- “sqlrowsread\_threshold\_violated - SQL rows read threshold violated monitor element” on page 651
- “sqlrowsreadinsc\_threshold\_id - SQL rows read in service class threshold ID monitor element” on page 652
- “sqlrowsreadinsc\_threshold\_value - SQL rows read in service class threshold value monitor element” on page 652
- “sqlrowsreadinsc\_threshold\_violated - SQL rows read in service class threshold violated monitor element” on page 652
- “sqlrowsreturned\_threshold\_id - SQL rows read returned threshold ID monitor element” on page 653
- “sqlrowsreturned\_threshold\_value - SQL rows read returned threshold value monitor element” on page 653
- “sqlrowsreturned\_threshold\_violated - SQL rows read returned threshold violated monitor element” on page 653
- “sqltempespace\_threshold\_id - SQL temporary space threshold ID monitor element” on page 654
- “sqltempespace\_threshold\_value - SQL temporary space threshold value monitor element” on page 654
- “sqltempespace\_threshold\_violated - SQL temporary space threshold violated monitor element” on page 654
- “stmt\_pkgcache\_id - Statement package cache identifier” on page 666
- “stmt\_text - SQL statement text monitor element” on page 669
- “tab\_file\_id - Table file ID monitor element” on page 677
- “tab\_type - Table type monitor element” on page 677
- “table\_file\_id - Table file ID monitor element” on page 678
- “table\_name - Table name monitor element” on page 678
- “table\_scans - Table scans monitor element” on page 679
- “table\_schema - Table schema name monitor element” on page 679
- “table\_type - Table type monitor element” on page 681
- “tablespace\_auto\_resize\_enabled - Table space automatic resizing enabled monitor element” on page 681
- “tablespace\_content\_type - Table space content type monitor element” on page 682
- “tablespace\_cur\_pool\_id - Buffer pool currently being used monitor element” on page 682
- “tablespace\_extent\_size - Table space extent size monitor element” on page 683
- “tablespace\_free\_pages - Free pages in table space monitor element” on page 683
- “tablespace\_id - Table space identification monitor element” on page 684
- “tablespace\_name - Table space name monitor element” on page 687
- “tablespace\_next\_pool\_id - Buffer pool that will be used at next startup monitor element” on page 688



- “tablespace\_page\_size - Table space page size monitor element” on page 689
- “tablespace\_page\_top - Table space high watermark monitor element” on page 689
- “tablespace\_paths\_dropped - Table space using dropped path monitor element” on page 689
- “tablespace\_pending\_free\_pages - Pending free pages in table space monitor element” on page 690
- “tablespace\_prefetch\_size - Table space prefetch size monitor element” on page 690
- “tablespace\_rebalancer\_mode - Rebalancer mode monitor element” on page 692
- “tablespace\_state - Table space state monitor element” on page 694
- “tablespace\_total\_pages - Total pages in table space monitor element” on page 696
- “tablespace\_type - Table space type monitor element” on page 696
- “tablespace\_usable\_pages - Usable pages in table space monitor element” on page 697
- “tablespace\_used\_pages - Used pages in table space monitor element” on page 697
- “tablespace\_using\_auto\_storage - Table space enabled for automatic storage monitor element” on page 698
- “tbsp\_max\_page\_top - Maximum table space page high watermark monitor element” on page 698
- “tcpip\_rcv\_volume - TCP/IP received volume monitor element” on page 698
- “tcpip\_rcv\_wait\_time - TCP/IP received wait time monitor element” on page 699
- “tcpip\_recvs\_total - TCP/IP receives total monitor element” on page 700
- “tcpip\_send\_volume - TCP/IP send volume monitor element” on page 701
- “tcpip\_send\_wait\_time - TCP/IP send wait time monitor element” on page 702
- “tcpip\_sends\_total - TCP/IP sends total monitor element” on page 702
- “total\_act\_time - Total activity time monitor element” on page 709
- “total\_act\_wait\_time - Total activity wait time monitor element” on page 709
- “total\_app\_rqst\_time - Total application request time monitor element” on page 710
- “total\_cpu\_time - Total CPU time monitor element” on page 712
- “total\_move\_time - Total extent move time monitor element” on page 713
- “total\_rqst\_mapped\_in - Total request mapped-in monitor element” on page 716
- “total\_rqst\_mapped\_out - Total request mapped-out monitor element” on page 716
- “total\_rqst\_time - Total request time monitor element” on page 716
- “total\_section\_sorts - Total section sorts monitor element” on page 717
- “total\_section\_sort\_proc\_time - Total section sort processing time monitor element” on page 718
- “total\_section\_sort\_time - Total section sort time monitor element” on page 719
- “total\_sorts - Total sorts monitor element” on page 721
- “total\_wait\_time - Total wait time monitor element” on page 725
- “tq\_tot\_send\_spills - Total number of table queue buffers overflowed monitor element” on page 730
- “unread\_prefetch\_pages - Unread prefetch pages monitor element” on page 733



- “uow\_id - Unit of work ID monitor element” on page 734
- “utility\_id - Utility ID” on page 740
- “valid - Section validity indicator monitor element” on page 741
- “vectored\_ios - Number of vectored I/O requests monitor element” on page 742
- “wlm\_queue\_assignments\_total - Workload manager total queue assignments monitor element” on page 742
- “wlm\_queue\_time\_total - Workload manager total queue time monitor element” on page 743
- “workload\_id - Workload ID monitor element” on page 745
- “workload\_name - Workload name monitor element” on page 746
- “workload\_occurrence\_id - Workload occurrence identifier monitor element” on page 747
- “workload\_occurrence\_state - Workload occurrence state monitor element” on page 748

---

## Chapter 8. Request monitor elements

Use request monitor elements to monitor the database system, specifically the volume of work and the effort expended by the data server to process application requests.

A request is a directive to a database agent to perform some work that expends database resources. Sources of the request can include:

- A directive issued directly by an external application, such as an OPEN or EXECUTE directive. These are referred to as application requests.
- A directive issued by a coordinator agent to a subagent at the same or a different database member.
- A directive issued by an agent at a different database member.

Request monitor elements measure the volume of work or effort expended by the database server to process different types of requests, including overall system processing, requests related to a specific type of processing, and requests related to a specific data server environment.

Some representative monitor elements for measuring overall system processing information are the following:

- The **rqsts\_completed\_total** monitor element measures the number of completed by the system.
- The **total\_rqst\_time** monitor element measures the time spent by requests in the data server, including wait time and processing time
- The **total\_wait\_time** monitor element measures the overall wait time.
- The **total\_cpu\_time** monitor element measures the CPU usage time.

Some representative monitor elements for measuring client-server processing information are the following:

- The **client\_idle\_wait\_time** monitor element measures the time spent waiting for the next request from an open connection.
- The **tcpip\_rcv\_volume** monitor element measures the volume of data received by the data server from clients over TCP/IP.

Some representative monitor elements for measuring common data server processing operations are the following:

- **pool\_data\_l\_reads** is one of the monitor elements providing information about buffer pool resource usage.
- **pool\_read\_time** is one of the monitor elements providing information about I/O processing.
- **lock\_wait\_time** is one of the monitor elements providing information about locks and locking.
- **total\_section\_sorts** is one of the monitor elements providing information about sorts.

Some representative monitor elements for monitoring processing relevant to selected types of data server environments are the following:

- **fcm\_rcv\_wait\_time** is one of the monitor elements measuring fast communications manager (FCM) processing.

- **wlm\_queue\_time\_total** is one of the monitor elements measuring workload management control actions.

## Accessing request metrics using table functions

You can use the following table functions to access the request metrics:

- `MON_GET_SERVICE_SUBCLASS` and `MON_GET_SERVICE_SUBCLASS_DETAILS`
- `MON_GET_WORKLOAD` and `MON_GET_WORKLOAD_DETAILS`
- `MON_GET_CONNECTION` and `MON_GET_CONNECTION_DETAILS`
- `MON_GET_UNIT_OF_WORK` and `MON_GET_UNIT_OF_WORK_DETAILS`

Each table function in this set of monitor table functions has two forms, one of which has a name ending with "DETAILS." The function that does not end with "DETAILS" provides an SQL relational interface that returns the most commonly needed data. The other function provides XML-based access to the monitor data and returns a more comprehensive set of data.

This set of table functions enables you to focus on request metrics at a particular level of aggregation. You can choose the table function that enables you to focus on subset (or aggregation) of the system workload you are interested in a given situation. All of these table functions include a common set of request metric monitor elements. Each table function may return a few additional details not common with all the table functions.

In a database with no user-defined workloads or service classes, all of the user work performed by the database manager occurs in the default user workload and user service class. The table functions that return data for each service class (or workload) return data for a single service class (or workload) that represents the processing for the user workload for the entire database.

In a database with user-defined workloads and service classes, table functions that return data for each service class (or workload) enable you to compare processing per service class (or workload). Using SQL, you can sum the values across service classes (or workloads) to obtain the value of a monitor element that represents the processing for the user workload for the entire database.

## Accessing request metrics using event monitors

Request metrics are reported by the following event monitors:

- Statistics event monitor - Request metrics are one of several types of information reported by this event monitor.
- UoW event monitor - This event monitor reports similar or identical fields as the `MON_GET_UNIT_OF_WORK` table function

---

## Chapter 9. Activity monitor elements

Activity monitor elements are a subset of request monitor elements. Use activity metrics to monitor the subset of data server processing related to executing activities, especially processing done to execute SQL statement sections.

Request monitor elements monitor the complete volume of work and effort expended by the data server to process application requests. Activity monitor elements monitor the work done to execute SQL statement sections, including locking, sorting, and row processing.

To access the current values for activity monitor elements, use the following table functions:

### **MON\_GET\_ACTIVITY\_DETAILS**

Returns details about one or more activities in progress. Specify the activities of interest in the input parameters. Data returned includes activity metric monitor elements, many other monitor elements, and statement text. Data is returned in XML format.

### **MON\_GET\_PKG\_CACHE\_STMT**

Returns details for some or all SQL statement sections in the database package cache, which includes both static and dynamic SQL statements. Data returned includes activity metric monitor elements aggregated over all executions of the section since it was added to the package cache. Data is returned in a relational form.

Use the activity event monitor to access historical data about activities. This monitor captures data for each execution of each activity. The activity event monitor captures the same activity monitor elements as the `MON_GET_ACTIVITY_DETAILS` table function. It also captures some additional information.



---

## Chapter 10. Data object monitor elements

Data object monitor elements provide information about operations performed on particular data objects, including tables, indexes, buffer pools, table spaces, and containers.

Every data object type has a set of monitor elements that can be monitored. For example, buffer pools have elements that can be used to calculate buffer pool hit ratios.

Use the following table functions to access current values for data object monitor elements. These monitor table functions return data in a relational form:

- MON\_GET\_BUFFERPOOL
- MON\_GET\_TABLESPACE
- MON\_GET\_CONTAINER
- MON\_GET\_TABLE
- MON\_GET\_INDEX



---

## Chapter 11. Monitor elements reported by the unit of work event monitor

The following monitor elements are reported by the unit of work event monitor.

- “agent\_id - Application handle (agent ID) monitor element” on page 313
- “appl\_id - Application ID” on page 325
- “appl\_name - Application name monitor element” on page 328
- “auth\_id - Authorization ID” on page 340
- “client\_acctng - Client accounting string monitor element” on page 353
- “client\_applname - Client application name monitor element” on page 354
- “client\_pid - Client Process ID” on page 356
- “client\_platform - Client Operating Platform” on page 357
- “client\_prdid - Client product and version ID monitor element” on page 357
- “client\_protocol - Client Communication Protocol” on page 358
- “client\_userid - Client user ID monitor element” on page 358
- “client\_wrkstnname - Client workstation name monitor element” on page 359
- “completion\_status - Completion status monitor element” on page 362
- “conn\_time - Time of database connection monitor element” on page 371
- “coord\_partition\_num - Coordinator partition number monitor element” on page 383
- “db\_conn\_time - Database activation timestamp monitor element” on page 392
- “service\_class\_id - Service class ID monitor element” on page 639
- “service\_subclass\_name - Service subclass name monitor element” on page 640
- “service\_superclass\_name - Service superclass name monitor element” on page 640
- “session\_auth\_id - Session authorization ID monitor element” on page 641
- “uow\_id - Unit of work ID monitor element” on page 734
- “uow\_start\_time - Unit of Work Start Timestamp” on page 736
- “uow\_stop\_time - Unit of work stop timestamp monitor element” on page 737
- “workload\_id - Workload ID monitor element” on page 745
- “workload\_name - Workload name monitor element” on page 746
- “workload\_occurrence\_id - Workload occurrence identifier monitor element” on page 747





---

## Chapter 12. Monitor elements reported by the locking event monitor

These monitor elements are reported by the locking event monitor.

- “activity\_id - Activity ID monitor element” on page 309
- “agent\_id - Application handle (agent ID) monitor element” on page 313
- “agent\_pid - Engine dispatchable unit (EDU) identifier monitor element” on page 315
- “appl\_id - Application ID” on page 325
- “appl\_name - Application name monitor element” on page 328
- “appl\_status - Application Status” on page 331
- “auth\_id - Authorization ID” on page 340
- “client\_acctng - Client accounting string monitor element” on page 353
- “client\_applname - Client application name monitor element” on page 354
- “client\_userid - Client user ID monitor element” on page 358
- “client\_wrkstnname - Client workstation name monitor element” on page 359
- “consistency\_token - Package consistency token monitor element” on page 373
- “coord\_agent\_pid - Coordinator agent identifier monitor element” on page 382
- “dl\_conns - Connections involved in deadlock monitor element” on page 414
- “effective\_isolation - Effective isolation monitor element” on page 415
- “effective\_query\_degree - Effective query degree monitor element” on page 416
- “lock\_attributes - Lock attributes monitor element” on page 477
- “lock\_count - Lock count monitor element” on page 478
- “lock\_current\_mode - Original Lock Mode Before Conversion” on page 479
- “lock\_escalation - Lock escalation monitor element” on page 480
- “lock\_hold\_count - Lock hold count monitor element” on page 482
- “lock\_mode - Lock mode monitor element” on page 483
- “lock\_mode\_requested - Lock mode requested monitor element” on page 484
- “lock\_name - Lock name monitor element” on page 484
- “lock\_object\_type - Lock object type waited on monitor element” on page 486
- “lock\_release\_flags - Lock release flags monitor element” on page 486
- “lock\_status - Lock status monitor element” on page 487
- “lock\_timeout\_val - Lock timeout value monitor element” on page 488
- “package\_name - Package name monitor element” on page 534
- “package\_schema - Package schema monitor element” on page 535
- “package\_version\_id - Package version monitor element” on page 535
- “rolled\_back\_participant\_no - Rolled back application participant monitor element” on page 620
- “section\_number - Section number monitor element” on page 633
- “service\_class\_id - Service class ID monitor element” on page 639
- “service\_subclass\_name - Service subclass name monitor element” on page 640
- “stmt\_invocation\_id - Statement invocation identifier monitor element” on page 662
- “stmt\_lock\_timeout - Statement lock timeout monitor element” on page 663

- “stmt\_nest\_level - Statement nesting level monitor element” on page 664
- “stmt\_operation/operation - Statement operation monitor element” on page 664
- “stmt\_pkgcache\_id - Statement package cache identifier” on page 666
- “stmt\_query\_id - Statement query identifier monitor element” on page 666
- “stmt\_source\_id - Statement source identifier” on page 667
- “stmt\_text - SQL statement text monitor element” on page 669
- “stmt\_type - Statement type monitor element” on page 670
- “stmt\_value\_data - Value data” on page 672
- “stmt\_value\_index - Value index” on page 672
- “stmt\_value\_isnull - Value has null value monitor element” on page 672
- “stmt\_value\_isreopt - Variable used for statement reoptimization monitor element” on page 673
- “stmt\_value\_type - Value type monitor element” on page 673
- “table\_name - Table name monitor element” on page 678
- “table\_schema - Table schema name monitor element” on page 679
- “tablespace\_name - Table space name monitor element” on page 687
- “uow\_id - Unit of work ID monitor element” on page 734
- “workload\_id - Workload ID monitor element” on page 745
- “workload\_name - Workload name monitor element” on page 746

## Chapter 13. Wait time monitor elements

Finding if and where requests are spending time waiting during request processing is an effective way to find the location of a performance slow down or a bottleneck. Wait time monitor elements measure the time spent waiting for a particular entity or resource.

For example, a common explanation for performance slow downs is that the requests are spending too much time waiting for a particular lock. To determine the nature of the delay, you might look at the values for the **lock\_wait\_time** monitor element.

Each wait time monitor element also belongs in the set of request monitor elements, activity monitor elements, or data object monitor elements.

Table 70. Wait time monitor elements

Category	Sub category	Wait time monitor element
High level wait times	Wait time during DB2 processing, including activity execution	"total_wait_time - Total wait time monitor element" on page 725
	Wait time during activity execution only	"total_act_wait_time - Total activity wait time monitor element" on page 709
I/O	Buffer pool I/O	"pool_read_time - Total buffer pool physical read time monitor element" on page 571 "pool_write_time - Total buffer pool physical write time monitor element" on page 584
	Direct I/O for LOBs	"direct_read_time - Direct read time monitor element" on page 406 "direct_write_time - Direct write time monitor element" on page 410
	Transaction logging I/O	"log_buffer_wait_time - Log buffer wait time monitor element" on page 495
	Diagnostic message logging	"diaglog_write_wait_time - Diagnostic log file write wait time monitor element" on page 402
Locking	-	"lock_wait_time - Time waited on locks monitor element" on page 490
Connect	Agent wait time	"agent_wait_time - Agent wait time monitor element" on page 316
Audit	-	"audit_file_write_wait_time - Audit file write wait time monitor element" on page 336 "audit_subsystem_wait_time - Audit subsystem wait time monitor element" on page 338

Table 70. Wait time monitor elements (continued)

Category	Sub category	Wait time monitor element
FCM	FCM overall (request messages and table queue data)	<p>"fcm_send_wait_time - FCM send wait time monitor element" on page 430</p> <p>"fcm_rcv_wait_time - FCM received wait time monitor element" on page 427</p>
	FCM related to request messages	<p>"fcm_message_send_wait_time - FCM message send wait time monitor element" on page 425</p> <p>"fcm_message_rcv_wait_time - FCM message received wait time monitor element" on page 423</p>
	FCM related to table queues	<p>"fcm_tq_send_wait_time - FCM table queue send wait time monitor element" on page 436</p> <p>"fcm_tq_rcv_wait_time - FCM table queue received wait time monitor element" on page 433</p>
Workload manager control actions	Queuing due to concurrency threshold exceeded	"wlm_queue_time_total - Workload manager total queue time monitor element" on page 743
Client-server processing	Time spent waiting for the next request from an active connection	"client_idle_wait_time - Client idle wait time monitor element" on page 355
	Remote client network (TCPIP)	<p>"tcpip_rcv_wait_time - TCP/IP received wait time monitor element" on page 699</p> <p>"tcpip_send_wait_time - TCP/IP send wait time monitor element" on page 702</p>
	Local client communication (IPC)	<p>"ipc_rcv_wait_time - Interprocess communication received wait time monitor element" on page 469</p> <p>"ipc_send_wait_time - Interprocess communication send wait time monitor element" on page 471</p>

The wait time elements are available through the following interfaces:

- MON\_GET\_SERVICE\_SUBCLASS table function
- MON\_GET\_SERVICE\_SUBCLASS\_DETAILS table function
- MON\_GET\_WORKLOAD table function
- MON\_GET\_WORKLOAD\_DETAILS table function
- MON\_GET\_CONNECTION table function
- MON\_GET\_CONNECTION\_DETAILS table function
- MON\_GET\_UNIT\_OF\_WORK table function
- MON\_GET\_UNIT\_OF\_WORK\_DETAILS table function
- MON\_GET\_PKG\_CACHE\_STMT table function
- MON\_GET\_ACTIVITY\_DETAILS table function

- Statistics event monitor (DETAILS\_XML element in the event\_wlstats and event\_scstats logical groups)
- Activity event monitor (DETAILS\_XML element in the event\_activity logical group)
- Unit of work event monitor

Not all wait time elements are reported through all interfaces. For example, the **client\_idle\_wait\_time** monitor element is only applicable to system-level interfaces like the MON\_GET\_SERVICE\_SUBCLASS table function. It is not reported by data object monitor table functions.

Refer to the description of each monitor element for a list of the interfaces that report the element.



---

## Chapter 14. Logical data groups

---

### Snapshot monitor interface mappings to logical data groups

The following table lists several ways of accessing snapshot monitor data. All snapshot monitor data is stored in monitor elements, which are categorized by logical data groups. Each individual API request type, CLP command, and SQL administrative view only captures monitor data from a subset of all the logical data groups.

Each individual API request type, CLP command, and SQL administrative view listed in this table returns monitor elements from the logical data groups listed in the right-most column.

**Note:**

1. There are a number of API request types and CLP commands for which there are no corresponding SQL administrative view. For other API request types and CLP commands, individual SQL administrative views capture subsets of the associated logical data groups.
2. Some monitor elements are returned only if the associated monitor switch is set ON. See the individual monitor elements to determine if a required element is under switch control.

*Table 71. Snapshot Monitor Interface Mappings to Logical Data Groups*

db2GetSnapshot API request type	CLP command	SQL administrative view	Logical data groups
SQLMA_APPLINFO_ALL	list applications [show detail]	APPLICATIONS	appl_info
SQLMA_DBASE_APPLINFO	list applications for database <i>dbname</i> [show detail]	APPLICATIONS	appl_info
SQLMA_DCS_APPLINFO_ALL	list dcs applications [show detail]		dcs_appl_info
SQLMA_DB2	get snapshot for dbm	SNAPDBM	db2
		SNAPFCM	fcm
		SNAPFCMPART	fcm_node
		SNAPUTIL	utility_info
		SNAPUTIL_PROGRESS	progress, progress_info
	SNAPDBM_MEMORY_POOL	memory_pool	
	get dbm monitor switches	SNAPSWITCHES	switch_list



Table 71. Snapshot Monitor Interface Mappings to Logical Data Groups (continued)

db2GetSnapshot API request type	CLP command	SQL administrative view	Logical data groups
SQLMA_DBASE	get snapshot for database on <i>dbname</i>	SNAPDB	dbase
		SNAPDETAILLOG	detail_log
		SNAPSTORAGE_PATHS	db_storage_group
			rollforward
			db_sto_path_info
		SNAPTbsp	tablespace
		SNAPDB_MEMORY_POOL	memory_pool
SQLMA_DBASE_ALL	get snapshot for all databases	SNAPDB	dbase
		SNAPSTORAGE_PATHS	db_storage_group
			rollforward
			db_sto_path_info
		SNAPTbsp	tablespace
		SNAPDB_MEMORY_POOL	memory_pool
	list active databases	dbase	
SQLMA_DCS_DBASE	get snapshot for dcs database on <i>dbname</i>		dcs_dbase, stmt_transmissions
SQLMA_DCS_DBASE_ALL	get snapshot for all dcs databases		dcs_dbase, stmt_transmissions
SQLMA_DBASE_REMOTE	get snapshot for remote database on <i>dbname</i>		dbase_remote
SQLMA_DBASE_REMOTE_ALL	get snapshot for all remote databases		dbase_remote
SQLMA_APPL	get snapshot for application applid <i>appl-id</i>	SNAPAPPL	appl
		SNAPAGENT	agent
		SNAPAPPL_INFO	appl_info
		SNAPLOCKWAIT	lock_wait
		SNAPSTMT	stmt
		SNAPSUBSECTION	subsection
		SNAPAGENT_MEMORY_POOL	memory_pool
SQLMA_AGENT_ID	get snapshot for application agentid <i>appl-handle</i>	SNAPAGENT	appl
		SNAPAGENT	agent
		SNAPAPPL_INFO	appl_info
		SNAPLOCKWAIT	lock_wait
		SNAPSTMT	stmt
		SNAPSUBSECTION	subsection
		SNAPAGENT_MEMORY_POOL	memory pool

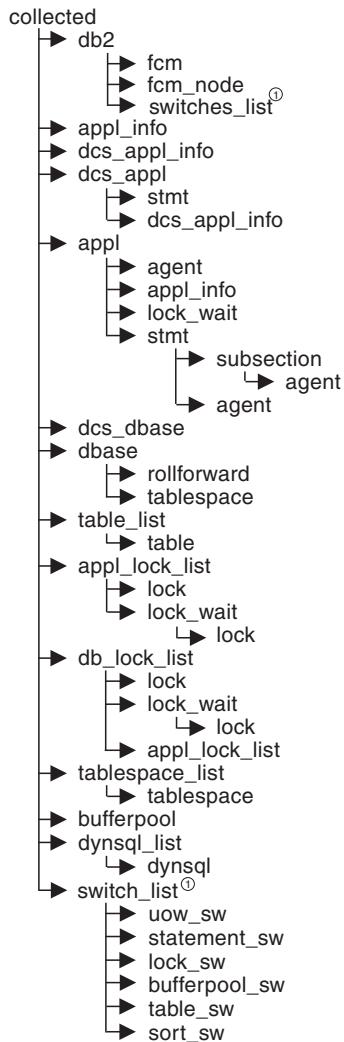
Table 71. Snapshot Monitor Interface Mappings to Logical Data Groups (continued)

db2GetSnapshot API request type	CLP command	SQL administrative view	Logical data groups
SQLMA_DBASE_APPLS	get snapshot for applications on <i>dbname</i>	SNAPAPPL	appl
		SNAPAGENT	agent
		SNAPAPPL_INFO	appl_info
		SNAPLOCKWAIT	lock_wait
		SNAPSTMT	stmt
		SNAPSUBSECTION	subsection
		SNAPAGENT_MEMORY_POOL	memory_pool
SQLMA_APPL_ALL	get snapshot for all applications	SNAPAPPL	appl
		SNAPAPPL_INFO	appl_info
		SNAPLOCKWAIT	lock_wait
		SNAPSTATEMENT	stmt
		SNAPAGENT	agent
		SNAPSUBSECTION	subsection
		SNAPAGENT_MEMORY_POOL	memory_pool
SQLMA_DCS_APPL	get snapshot for dcs application applid <i>appl-id</i>		dc_s_appl, dc_s_stmt, dc_s_appl_info, dc_s_stmt_transmissions
SQLMA_DCS_APPL_ALL	get snapshot for all dcs applications		dc_s_appl, dc_s_stmt, dc_s_appl_info, dc_s_stmt_transmissions
SQLMA_DCS_APPL_HANDLE	get snapshot for dcs application agentid <i>appl-handle</i>		dc_s_appl, dc_s_stmt, dc_s_appl_info, dc_s_stmt_transmissions
SQLMA_DCS_DBASE_APPLS	get snapshot for dcs applications on <i>dbname</i>		dc_s_appl, dc_s_stmt, dc_s_appl_info, dc_s_stmt_transmissions
SQLMA_DBASE_APPLS_REMOTE	get snapshot for remote applications on <i>dbname</i>		dbase_appl
SQLMA_APPL_REMOTE_ALL	get snapshot for all remote applications		dbase_appl
SQLMA_DBASE_TABLES	get snapshot for tables on <i>dbname</i>	SNAPTAB	table
		SNAPTAB_REORG	table_reorg
			table_list
SQLMA_APPL_LOCKS	get snapshot for locks for application applid <i>appl-id</i>	SNAPLOCK, SNAPAPPL, SNAPLOCKWAIT	appl_lock_list, lock_wait, lock
SQLMA_APPL_LOCKS_AGENT_ID	get snapshot for locks for application agentid <i>appl-handle</i>	SNAPLOCK, SNAPAPPL, SNAPLOCKWAIT	appl_lock_list, lock_wait, lock

Table 71. Snapshot Monitor Interface Mappings to Logical Data Groups (continued)

db2GetSnapshot API request type	CLP command	SQL administrative view	Logical data groups
SQLMA_DBASE_LOCKS	get snapshot for locks on <i>dbname</i>	SNAPLOCK	appl_lock_list, lock
		SNAPLOCK, SNAPLOCKWAIT	db_lock_list, lock_wait
SQLMA_DBASE_TABLESPACES	get snapshot for tablespaces on <i>dbname</i>	SNAPTbsp	tablespace
		SNAPTbspPART	tablespace, tablespace_nodeinfo
		SNAPTbsp QUIESCER	tablespace_quiescer, tablespace_nodeinfo
		SNAPCONTAINER	tablespace_container, tablespace_nodeinfo
		SNAPTbsp_RANGE	tablespace_ranges, tablespace_nodeinfo
			tablespace_list, tablespace_nodeinfo
SQLMA_BUFFERPOOLS_ALL	get snapshot for all bufferpools	SNAPBP	bufferpool
SQLMA_DBASE_BUFFERPOOLS	get snapshot for bufferpools on <i>dbname</i>	SNAPBP	bufferpool
SQLMA_DYNAMIC_SQL	get snapshot for dynamic sql on <i>dbname</i>	SNAPDYN_SQL	dynsql
			dynsql_list

The following figure shows the order that logical data groupings may appear in a snapshot data stream.



① Similar structures (lower level\_sw items are returned by db2, but are not shown in the figure)

Figure 8. Data Stream Hierarchy

**Note:** Times may be returned as part of any logical data grouping.

## Snapshot monitor logical data groups and monitor elements

The following sections list the logical data groupings and monitor elements that can be returned by snapshot monitoring.

- “agent logical data group” on page 246
- “appl logical data group” on page 246
- “appl\_id\_info logical data group” on page 249
- “appl\_info logical data group” on page 250
- “appl\_lock\_list logical data group” on page 250
- “appl\_remote logical data group” on page 251
- “bufferpool logical data group” on page 251
- “bufferpool\_nodeinfo logical data group” on page 253
- “collected logical data group” on page 253
- “db2 logical data group” on page 253

- “db\_lock\_list logical data group” on page 254
- “dbase logical data group” on page 254
- “dbase\_remote logical data group” on page 259
- “db\_storage\_group logical data group” on page 259
- “dcs\_appl logical data group” on page 259
- “dcs\_appl\_info logical data group” on page 261
- “dcs\_dbase logical data group” on page 262
- “dcs\_stmt logical data group” on page 264
- “detail\_log logical data group” on page 264
- “dynsql logical data group” on page 264
- “dynsql\_list logical data group” on page 265
- “fcm logical data group” on page 266
- “fcm\_node logical data group” on page 266
- “hadr logical data group” on page 266
- “lock logical data group” on page 266
- “lock\_wait logical data group” on page 267
- “memory\_pool logical data group” on page 267
- “progress logical data group” on page 267
- “progress\_list logical data group” on page 268
- “rollforward logical data group” on page 268
- “stmt logical data group” on page 268
- “stmt\_transmissions logical data group” on page 269
- “subsection logical data group” on page 271
- “table logical data group” on page 271
- “table\_list logical data group” on page 271
- “table\_reorg logical data group” on page 272
- “tablespace logical data group” on page 272
- “tablespace\_container logical data group” on page 274
- “tablespace\_list logical data group” on page 274
- “tablespace\_nodeinfo logical data group” on page 274
- “tablespace\_quiescer logical data group” on page 275
- “tablespace\_range logical data group” on page 275
- “utility\_info logical data group” on page 276

### **agent logical data group**

“agent\_pid - Engine dispatchable unit (EDU) identifier monitor element” on page 315

“lock\_timeout\_val - Lock timeout value monitor element” on page 488

### **appl logical data group**

“acc\_curs\_blk - Accepted Block Cursor Requests” on page 302

“agent\_sys\_cpu\_time - System CPU Time used by Agent” on page 315

“agent\_usr\_cpu\_time - User CPU Time used by Agent” on page 316

“agents\_stolen - Stolen Agents” on page 320

“appl\_con\_time - Connection Request Start Timestamp” on page 324

“appl\_idle\_time - Application Idle Time” on page 328

“appl\_priority - Application Agent Priority” on page 329  
 “appl\_priority\_type - Application Priority Type” on page 329  
 “associated\_agents\_top - Maximum Number of Associated Agents” on page 335  
 “authority\_bitmap - User authorization level monitor element” on page 341  
 “authority\_lvl - User authorization level monitor element” on page 342  
 “binds\_precompiles - Binds/Precompiles Attempted” on page 344  
 “cat\_cache\_inserts - Catalog Cache Inserts” on page 349  
 “cat\_cache\_lookups - Catalog Cache Lookups” on page 350  
 “cat\_cache\_overflows - Catalog Cache Overflows” on page 351  
 “commit\_sql\_stmts - Commit Statements Attempted” on page 361  
 “conn\_complete\_time - Connection Request Completion Timestamp” on page 371  
 “ddl\_sql\_stmts - Data Definition Language (DDL) SQL Statements” on page 397  
 “deadlocks - Deadlocks detected monitor element” on page 399  
 “direct\_read\_reqs - Direct read requests monitor element” on page 404  
 “direct\_read\_time - Direct read time monitor element” on page 406  
 “direct\_reads - Direct reads from database monitor element” on page 407  
 “direct\_write\_reqs - Direct write requests monitor element” on page 409  
 “direct\_write\_time - Direct write time monitor element” on page 410  
 “direct\_writes - Direct writes to database monitor element” on page 412  
 “dynamic\_sql\_stmts - Dynamic SQL Statements Attempted” on page 414  
 “failed\_sql\_stmts - Failed Statement Operations” on page 421  
 “hash\_join\_overflows - Hash Join Overflows” on page 455  
 “hash\_join\_small\_overflows - Hash Join Small Overflows” on page 455  
 “inbound\_comm\_address - Inbound Communication Address” on page 459  
 “int\_auto\_rebinds - Internal Automatic Rebinds” on page 463  
 “int\_commits - Internal Commits” on page 463  
 “int\_deadlock\_rollbacks - Internal Rollbacks Due To Deadlock” on page 464  
 “int\_rollbacks - Internal Rollbacks” on page 465  
 “int\_rows\_deleted - Internal Rows Deleted” on page 466  
 “int\_rows\_inserted - Internal Rows Inserted” on page 466  
 “int\_rows\_updated - Internal Rows Updated” on page 467  
 “last\_reset - Last Reset Timestamp” on page 475  
 “lock\_escalation - Lock escalation monitor element” on page 480  
 “lock\_timeout\_val - Lock timeout value monitor element” on page 488  
 “lock\_timeouts - Number of lock timeouts monitor element” on page 488  
 “lock\_wait\_time - Time waited on locks monitor element” on page 490  
 “lock\_waits - Lock waits monitor element” on page 492  
 “locks\_held - Locks Held” on page 494  
 “locks\_waiting - Current Agents Waiting On Locks” on page 495  
 “num\_agents - Number of Agents Working on a Statement” on page 518  
 “olap\_func\_overflows - OLAP Function Overflows monitor element” on page 527  
 “open\_loc\_curs - Open Local Cursors” on page 528  
 “open\_loc\_curs\_blk - Open Local Cursors with Blocking” on page 528

“open\_rem\_curs - Open Remote Cursors” on page 528

“open\_rem\_curs\_blk - Open Remote Cursors with Blocking” on page 529

“pkg\_cache\_inserts - Package Cache Inserts” on page 543

“pkg\_cache\_lookups - Package Cache Lookups” on page 543

“pool\_data\_l\_reads - Buffer pool data logical reads monitor element” on page 556

“pool\_data\_p\_reads - Buffer pool data physical reads monitor element” on page 557

“pool\_data\_writes - Buffer pool data writes monitor element” on page 559

“pool\_index\_l\_reads - Buffer pool index logical reads monitor element” on page 564

“pool\_index\_p\_reads - Buffer pool index physical reads monitor element” on page 566

“pool\_index\_writes - Buffer pool index writes monitor element” on page 567

“pool\_read\_time - Total buffer pool physical read time monitor element” on page 571

“pool\_temp\_data\_l\_reads - Buffer pool temporary data logical reads monitor element” on page 573

“pool\_temp\_data\_p\_reads - Buffer pool temporary data physical reads monitor element” on page 575

“pool\_temp\_index\_l\_reads - Buffer pool temporary index logical reads monitor element” on page 577

“pool\_temp\_index\_p\_reads - Buffer pool temporary index physical reads monitor element” on page 578

“pool\_temp\_xda\_l\_reads - Buffer pool temporary XDA data logical reads monitor element” on page 580

“pool\_temp\_xda\_p\_reads - Buffer pool temporary XDA data physical reads monitor element” on page 582

“pool\_write\_time - Total buffer pool physical write time monitor element” on page 584

“pool\_xda\_l\_reads - Buffer pool XDA data logical reads monitor element” on page 585

“pool\_xda\_p\_reads - Buffer pool XDA data physical reads monitor element” on page 588

“pool\_xda\_writes - Buffer pool XDA data writes monitor element” on page 589

“prefetch\_wait\_time - Time waited for prefetch monitor element” on page 595

“prev\_uow\_stop\_time - Previous Unit of Work Completion Timestamp” on page 596

“priv\_workspace\_num\_overflows - Private Workspace Overflows” on page 597

“priv\_workspace\_section\_inserts - Private Workspace Section Inserts” on page 597

“priv\_workspace\_section\_lookups - Private Workspace Section Lookups” on page 598

“priv\_workspace\_size\_top - Maximum Private Workspace Size” on page 599

“rej\_curs\_blk - Rejected Block Cursor Requests” on page 609

“rollback\_sql\_stmts - Rollback Statements Attempted” on page 619

“rows\_deleted - Rows deleted monitor element” on page 621

“rows\_inserted - Rows inserted monitor element” on page 622

"rows\_read - Rows read monitor element" on page 624  
 "rows\_selected - Rows Selected" on page 628  
 "rows\_updated - Rows updated monitor element" on page 628  
 "rows\_written - Rows Written" on page 629  
 "select\_sql\_stmts - Select SQL Statements Executed" on page 634  
 "shr\_workspace\_num\_overflows - Shared Workspace Overflows" on page 642  
 "shr\_workspace\_section\_inserts - Shared Workspace Section Inserts" on page 642  
 "shr\_workspace\_section\_lookups - Shared Workspace Section Lookups" on page 643  
 "shr\_workspace\_size\_top - Maximum Shared Workspace Size" on page 643  
 "sort\_overflows - Sort overflows monitor element" on page 645  
 "sql\_reqs\_since\_commit - SQL Requests Since Last Commit" on page 649  
 "static\_sql\_stmts - Static SQL Statements Attempted" on page 657  
 "total\_hash\_joins - Total Hash Joins" on page 713  
 "total\_hash\_loops - Total Hash Loops" on page 714  
 "total\_olap\_funcs - Total OLAP Functions monitor element" on page 715  
 "total\_sort\_time - Total sort time monitor element" on page 720  
 "total\_sorts - Total sorts monitor element" on page 721  
 "uid\_sql\_stmts - Update/Insert/Delete SQL Statements Executed" on page 732  
 "unread\_prefetch\_pages - Unread prefetch pages monitor element" on page 733  
 "uow\_comp\_status - Unit of Work Completion Status" on page 733  
 "uow\_elapsed\_time - Most Recent Unit of Work Elapsed Time" on page 734  
 "uow\_lock\_wait\_time - Total time unit of work waited on locks monitor element" on page 735  
 "uow\_log\_space\_used - Unit of Work Log Space Used" on page 735  
 "uow\_start\_time - Unit of Work Start Timestamp" on page 736  
 "uow\_stop\_time - Unit of work stop timestamp monitor element" on page 737  
 "x\_lock\_escals - Exclusive Lock Escalations" on page 748  
 "xquery\_stmts - XQuery Statements Attempted" on page 750

### **appl\_id\_info logical data group**

"agent\_id - Application handle (agent ID) monitor element" on page 313  
 "appl\_id - Application ID" on page 325  
 "appl\_name - Application name monitor element" on page 328  
 "appl\_status - Application Status" on page 331  
 "auth\_id - Authorization ID" on page 340  
 "client\_db\_alias - Database Alias Used by Application" on page 355  
 "client\_prdid - Client product and version ID monitor element" on page 357  
 "codepage\_id - ID of Code Page Used by Application" on page 360  
 "db\_name - Database Name" on page 393  
 "db\_path - Database Path" on page 394  
 "input\_db\_alias - Input Database Alias" on page 461  
 "sequence\_no - Sequence number monitor element" on page 635  
 "status\_change\_time - Application Status Change Time" on page 660



## **appl\_info logical data group**

- "agent\_id - Application handle (agent ID) monitor element" on page 313
- "appl\_id - Application ID" on page 325
- "appl\_name - Application name monitor element" on page 328
- "appl\_section\_inserts - Section Inserts monitor element" on page 330
- "appl\_section\_lookups - Section Lookups" on page 330
- "appl\_status - Application Status" on page 331
- "auth\_id - Authorization ID" on page 340
- "authority\_bitmap - User authorization level monitor element" on page 341
- "authority\_lvl - User authorization level monitor element" on page 342
- "client\_db\_alias - Database Alias Used by Application" on page 355
- "client\_pid - Client Process ID" on page 356
- "client\_platform - Client Operating Platform" on page 357
- "client\_prdid - Client product and version ID monitor element" on page 357
- "client\_protocol - Client Communication Protocol" on page 358
- "codepage\_id - ID of Code Page Used by Application" on page 360
- "coord\_agent\_pid - Coordinator agent identifier monitor element" on page 382
- "coord\_node - Coordinating Node" on page 383
- "corr\_token - DRDA Correlation Token" on page 383
- "db\_name - Database Name" on page 393
- "db\_path - Database Path" on page 394
- "execution\_id - User Login ID" on page 421
- "input\_db\_alias - Input Database Alias" on page 461
- "is\_system\_appl - Is System Application monitor element" on page 473
- "num\_assoc\_agents - Number of Associated Agents" on page 519
- "sequence\_no - Sequence number monitor element" on page 635
- "session\_auth\_id - Session authorization ID monitor element" on page 641
- "status\_change\_time - Application Status Change Time" on page 660
- "territory\_code - Database Territory Code" on page 704
- "tpmon\_acc\_str - TP monitor client accounting string monitor element" on page 726
- "tpmon\_client\_app - TP monitor client application name monitor element" on page 726
- "tpmon\_client\_userid - TP monitor client user ID monitor element" on page 727
- "tpmon\_client\_wkstn - TP monitor client workstation name monitor element" on page 727
- "workload\_id - Workload ID monitor element" on page 745

## **appl\_lock\_list logical data group**

- "agent\_id - Application handle (agent ID) monitor element" on page 313
- "appl\_id - Application ID" on page 325
- "appl\_name - Application name monitor element" on page 328
- "appl\_status - Application Status" on page 331
- "auth\_id - Authorization ID" on page 340
- "client\_db\_alias - Database Alias Used by Application" on page 355
- "codepage\_id - ID of Code Page Used by Application" on page 360

*"lock\_wait\_time - Time waited on locks monitor element"* on page 490  
*"locks\_held - Locks Held"* on page 494  
*"locks\_waiting - Current Agents Waiting On Locks"* on page 495  
*"sequence\_no - Sequence number monitor element"* on page 635  
*"session\_auth\_id - Session authorization ID monitor element"* on page 641  
*"status\_change\_time - Application Status Change Time"* on page 660

### **appl\_remote logical data group**

*"commit\_sql\_stmts - Commit Statements Attempted"* on page 361  
*"create\_nickname - Create Nicknames"* on page 387  
*"create\_nickname\_time - Create Nickname Response Time"* on page 387  
*"datasource\_name - Data Source Name"* on page 391  
*"db\_name - Database Name"* on page 393  
*"delete\_sql\_stmts - Deletes"* on page 401  
*"delete\_time - Delete Response Time"* on page 402  
*"failed\_sql\_stmts - Failed Statement Operations"* on page 421  
*"insert\_sql\_stmts - Inserts"* on page 461  
*"insert\_time - Insert Response Time"* on page 462  
*"passthru\_time - Pass-Through Time"* on page 541  
*"passthru - Pass-Through"* on page 541  
*"remote\_lock\_time - Remote Lock Time"* on page 611  
*"remote\_locks - Remote Locks"* on page 611  
*"rollback\_sql\_stmts - Rollback Statements Attempted"* on page 619  
*"rows\_deleted - Rows deleted monitor element"* on page 621  
*"rows\_inserted - Rows inserted monitor element"* on page 622  
*"rows\_selected - Rows Selected"* on page 628  
*"rows\_updated - Rows updated monitor element"* on page 628  
*"select\_sql\_stmts - Select SQL Statements Executed"* on page 634  
*"select\_time - Query Response Time"* on page 634  
*"sp\_rows\_selected - Rows Returned by Stored Procedures"* on page 648  
*"stored\_proc\_time - Stored Procedure Time"* on page 675  
*"stored\_procs - Stored Procedures"* on page 675  
*"update\_sql\_stmts - Updates"* on page 738  
*"update\_time - Update Response Time"* on page 738

### **bufferpool logical data group**

*"block\_ios - Number of block I/O requests monitor element"* on page 344  
*"bp\_id - Buffer pool identifier monitor element"* on page 347  
*"bp\_name - Buffer pool name monitor element"* on page 347  
*"db\_name - Database Name"* on page 393  
*"db\_path - Database Path"* on page 394  
*"direct\_read\_reqs - Direct read requests monitor element"* on page 404  
*"direct\_read\_time - Direct read time monitor element"* on page 406  
*"direct\_reads - Direct reads from database monitor element"* on page 407  
*"direct\_write\_reqs - Direct write requests monitor element"* on page 409  
*"direct\_write\_time - Direct write time monitor element"* on page 410

“direct\_writes - Direct writes to database monitor element” on page 412

“files\_closed - Database files closed monitor element” on page 438

“input\_db\_alias - Input Database Alias” on page 461

“pages\_from\_block\_ios - Total number of pages read by block I/O monitor element” on page 536

“pages\_from\_vectored\_ios - Total number of pages read by vectored I/O monitor element” on page 537

“pool\_async\_data\_read\_reqs - Buffer pool asynchronous read requests monitor element” on page 546

“pool\_async\_data\_reads - Buffer pool asynchronous data reads monitor element” on page 547

“pool\_async\_data\_writes - Buffer pool asynchronous data writes monitor element” on page 547

“pool\_async\_index\_read\_reqs - Buffer pool asynchronous index read requests monitor element” on page 548

“pool\_async\_index\_reads - Buffer pool asynchronous index reads monitor element” on page 549

“pool\_async\_index\_writes - Buffer pool asynchronous index writes monitor element” on page 550

“pool\_async\_read\_time - Buffer Pool Asynchronous Read Time” on page 551

“pool\_async\_write\_time - Buffer Pool Asynchronous Write Time” on page 551

“pool\_async\_xda\_read\_reqs - Buffer pool asynchronous XDA read requests monitor element” on page 552

“pool\_async\_xda\_reads - Buffer pool asynchronous XDA data reads monitor element” on page 553

“pool\_async\_xda\_writes - Buffer pool asynchronous XDA data writes monitor element” on page 554

“pool\_data\_l\_reads - Buffer pool data logical reads monitor element” on page 556

“pool\_data\_p\_reads - Buffer pool data physical reads monitor element” on page 557

“pool\_data\_writes - Buffer pool data writes monitor element” on page 559

“pool\_index\_l\_reads - Buffer pool index logical reads monitor element” on page 564

“pool\_index\_p\_reads - Buffer pool index physical reads monitor element” on page 566

“pool\_index\_writes - Buffer pool index writes monitor element” on page 567

“pool\_no\_victim\_buffer - Buffer pool no victim buffers monitor element” on page 570

“pool\_read\_time - Total buffer pool physical read time monitor element” on page 571

“pool\_temp\_data\_l\_reads - Buffer pool temporary data logical reads monitor element” on page 573

“pool\_temp\_data\_p\_reads - Buffer pool temporary data physical reads monitor element” on page 575

“pool\_temp\_index\_l\_reads - Buffer pool temporary index logical reads monitor element” on page 577

“pool\_temp\_index\_p\_reads - Buffer pool temporary index physical reads monitor element” on page 578

"pool\_temp\_xda\_l\_reads - Buffer pool temporary XDA data logical reads monitor element" on page 580  
"pool\_temp\_xda\_p\_reads - Buffer pool temporary XDA data physical reads monitor element" on page 582  
"pool\_write\_time - Total buffer pool physical write time monitor element" on page 584  
"pool\_xda\_l\_reads - Buffer pool XDA data logical reads monitor element" on page 585  
"pool\_xda\_p\_reads - Buffer pool XDA data physical reads monitor element" on page 588  
"pool\_xda\_writes - Buffer pool XDA data writes monitor element" on page 589  
"vectored\_ios - Number of vectored I/O requests monitor element" on page 742

### **bufferpool\_nodeinfo logical data group**

"bp\_cur\_buffsz - Current Size of Buffer Pool" on page 346  
"bp\_new\_buffsz - New Buffer Pool Size" on page 347  
"bp\_pages\_left\_to\_remove - Number of Pages Left to Remove" on page 348  
"bp\_tbsp\_use\_count - Number of Table Spaces Mapped to Buffer Pool" on page 348  
"node\_number - Node Number" on page 517

### **collected logical data group**

"node\_number - Node Number" on page 517  
"server\_db2\_type - Database Manager Type at Monitored (Server) Node" on page 636  
"server\_instance\_name - Server Instance Name" on page 637  
"server\_prdid - Server Product/Version ID" on page 637  
"server\_version - Server Version" on page 638  
"time\_stamp - Snapshot Time" on page 708  
"time\_zone\_displacement - Time Zone Displacement" on page 708

### **db2 logical data group**

"agents\_created\_empty\_pool - Agents Created Due to Empty Agent Pool" on page 319  
"agents\_from\_pool - Agents Assigned From Pool" on page 319  
"agents\_registered - Agents Registered" on page 319  
"agents\_registered\_top - Maximum Number of Agents Registered" on page 320  
"agents\_stolen - Stolen Agents" on page 320  
"agents\_waiting\_on\_token - Agents Waiting for a Token" on page 321  
"agents\_waiting\_top - Maximum Number of Agents Waiting monitor element" on page 321  
"comm\_private\_mem - Committed Private Memory" on page 361  
"con\_local\_databases - Local Databases with Current Connects" on page 363  
"coord\_agents\_top - Maximum Number of Coordinating Agents" on page 382  
"db2start\_time - Start Database Manager Timestamp" on page 392  
"db\_status - Status of Database" on page 394  
"gw\_cons\_wait\_client - Number of Connections Waiting for the Client to Send Request" on page 443

"gw\_cons\_wait\_host - Number of Connections Waiting for the Host to Reply" on page 444  
 "gw\_cur\_cons - Current Number of Connections for DB2 Connect" on page 444  
 "gw\_total\_cons - Total Number of Attempted Connections for DB2 Connect" on page 445  
 "idle\_agents - Number of Idle Agents" on page 458  
 "last\_reset - Last Reset Timestamp" on page 475  
 "local\_cons - Local Connections" on page 476  
 "local\_cons\_in\_exec - Local Connections Executing in the Database Manager" on page 477  
 "max\_agent\_overflows - Maximum Agent Overflows" on page 502  
 "num\_gw\_conn\_switches - Connection Switches" on page 521  
 "num\_nodes\_in\_db2\_instance - Number of Nodes in Partition" on page 524  
 "piped\_sorts\_accepted - Piped Sorts Accepted" on page 542  
 "piped\_sorts\_requested - Piped Sorts Requested" on page 542  
 "post\_threshold\_hash\_joins - Hash Join Threshold" on page 593  
 "post\_threshold\_olap\_funcs - OLAP Function Threshold monitor element" on page 593  
 "post\_threshold\_sorts - Post threshold sorts monitor element" on page 593  
 "product\_name - Product Name" on page 599  
 "rem\_cons\_in - Remote Connections To Database Manager" on page 610  
 "rem\_cons\_in\_exec - Remote Connections Executing in the Database Manager" on page 610  
 "service\_level - Service Level" on page 639  
 "smallest\_log\_avail\_node - Node with Least Available Log Space" on page 644  
 "sort\_heap\_allocated - Total Sort Heap Allocated" on page 644  
 "sort\_heap\_top - Sort private heap high watermark" on page 645

### **db\_lock\_list logical data group**

"appls\_cur\_cons - Applications Connected Currently" on page 334  
 "db\_name - Database Name" on page 393  
 "db\_path - Database Path" on page 394  
 "input\_db\_alias - Input Database Alias" on page 461  
 "locks\_held - Locks Held" on page 494  
 "locks\_waiting - Current Agents Waiting On Locks" on page 495

### **dbase logical data group**

"active\_hash\_joins - Active hash joins" on page 308  
 "active\_olap\_funcs - Active OLAP Functions monitor element" on page 308  
 "active\_sorts - Active Sorts" on page 308  
 "agents\_top - Number of Agents Created" on page 321  
 "appl\_id\_oldest\_xact - Application with Oldest Transaction" on page 327  
 "appl\_section\_inserts - Section Inserts monitor element" on page 330  
 "appl\_section\_lookups - Section Lookups" on page 330  
 "appls\_cur\_cons - Applications Connected Currently" on page 334  
 "appls\_in\_db2 - Applications Executing in the Database Currently" on page 334

"async\_runstats - Total number of asynchronous RUNSTATS requests monitor element" on page 335

"binds\_precompiles - Binds/Precompiles Attempted" on page 344

"blocks\_pending\_cleanup - Pending cleanup rolled-out blocks monitor element" on page 346

"cat\_cache\_inserts - Catalog Cache Inserts" on page 349

"cat\_cache\_lookups - Catalog Cache Lookups" on page 350

"cat\_cache\_overflows - Catalog Cache Overflows" on page 351

"cat\_cache\_size\_top - Catalog cache high watermark monitor element" on page 351

"catalog\_node - Catalog Node Number" on page 352

"catalog\_node\_name - Catalog Node Network Name" on page 352

"commit\_sql\_stmts - Commit Statements Attempted" on page 361

"connections\_top - Maximum Number of Concurrent Connections" on page 372

"coord\_agents\_top - Maximum Number of Coordinating Agents" on page 382

"db\_conn\_time - Database activation timestamp monitor element" on page 392

"db\_heap\_top - Maximum Database Heap Allocated" on page 392

"db\_location - Database Location" on page 393

"db\_name - Database Name" on page 393

"db\_path - Database Path" on page 394

"db\_status - Status of Database" on page 394

"ddl\_sql\_stmts - Data Definition Language (DDL) SQL Statements" on page 397

"deadlocks - Deadlocks detected monitor element" on page 399

"direct\_read\_reqs - Direct read requests monitor element" on page 404

"direct\_read\_time - Direct read time monitor element" on page 406

"direct\_reads - Direct reads from database monitor element" on page 407

"direct\_write\_reqs - Direct write requests monitor element" on page 409

"direct\_write\_time - Direct write time monitor element" on page 410

"direct\_writes - Direct writes to database monitor element" on page 412

"dynamic\_sql\_stmts - Dynamic SQL Statements Attempted" on page 414

"failed\_sql\_stmts - Failed Statement Operations" on page 421

"files\_closed - Database files closed monitor element" on page 438

"hash\_join\_overflows - Hash Join Overflows" on page 455

"hash\_join\_small\_overflows - Hash Join Small Overflows" on page 455

"input\_db\_alias - Input Database Alias" on page 461

"int\_auto\_rebinds - Internal Automatic Rebinds" on page 463

"int\_commits - Internal Commits" on page 463

"int\_deadlock\_rollbacks - Internal Rollbacks Due To Deadlock" on page 464

"int\_rollbacks - Internal Rollbacks" on page 465

"int\_rows\_deleted - Internal Rows Deleted" on page 466

"int\_rows\_inserted - Internal Rows Inserted" on page 466

"int\_rows\_updated - Internal Rows Updated" on page 467

"last\_backup - Last Backup Timestamp" on page 474

"last\_reset - Last Reset Timestamp" on page 475

"lock\_escals - Number of lock escalations monitor element" on page 480

"lock\_list\_in\_use - Total Lock List Memory In Use" on page 482  
"lock\_timeouts - Number of lock timeouts monitor element" on page 488  
"lock\_wait\_time - Time waited on locks monitor element" on page 490  
"lock\_waits - Lock waits monitor element" on page 492  
"locks\_held - Locks Held" on page 494  
"locks\_waiting - Current Agents Waiting On Locks" on page 495  
"log\_held\_by\_dirty\_pages - Amount of Log Space Accounted for by Dirty Pages" on page 498  
"log\_read\_time - Log Read Time" on page 499  
"log\_reads - Number of Log Pages Read" on page 499  
"log\_to\_redo\_for\_recovery - Amount of Log to be Redone for Recovery" on page 500  
"log\_write\_time - Log Write Time" on page 500  
"log\_writes - Number of Log Pages Written" on page 500  
"num\_assoc\_agents - Number of Associated Agents" on page 519  
"num\_db\_storage\_paths - Number of automatic storage paths" on page 519  
"num\_indoubt\_trans - Number of Indoubt Transactions" on page 521  
"num\_log\_buffer\_full - Number of full log buffers monitor element" on page 522  
"num\_log\_data\_found\_in\_buffer - Number of Log Data Found In Buffer" on page 523  
"num\_log\_part\_page\_io - Number of Partial Log Page Writes" on page 523  
"num\_log\_read\_io - Number of Log Reads" on page 524  
"num\_log\_write\_io - Number of Log Writes" on page 524  
"olap\_func\_overflows - OLAP Function Overflows monitor element" on page 527  
"pkg\_cache\_inserts - Package Cache Inserts" on page 543  
"pkg\_cache\_lookups - Package Cache Lookups" on page 543  
"pkg\_cache\_num\_overflows - Package Cache Overflows" on page 545  
"pkg\_cache\_size\_top - Package cache high watermark" on page 545  
"pool\_async\_data\_read\_reqs - Buffer pool asynchronous read requests monitor element" on page 546  
"pool\_async\_data\_reads - Buffer pool asynchronous data reads monitor element" on page 547  
"pool\_async\_data\_writes - Buffer pool asynchronous data writes monitor element" on page 547  
"pool\_async\_index\_read\_reqs - Buffer pool asynchronous index read requests monitor element" on page 548  
"pool\_async\_index\_reads - Buffer pool asynchronous index reads monitor element" on page 549  
"pool\_async\_index\_writes - Buffer pool asynchronous index writes monitor element" on page 550  
"pool\_async\_read\_time - Buffer Pool Asynchronous Read Time" on page 551  
"pool\_async\_write\_time - Buffer Pool Asynchronous Write Time" on page 551  
"pool\_async\_xda\_read\_reqs - Buffer pool asynchronous XDA read requests monitor element" on page 552



“pool\_async\_xda\_reads - Buffer pool asynchronous XDA data reads monitor element” on page 553

“pool\_async\_xda\_writes - Buffer pool asynchronous XDA data writes monitor element” on page 554

“pool\_data\_l\_reads - Buffer pool data logical reads monitor element” on page 556

“pool\_data\_p\_reads - Buffer pool data physical reads monitor element” on page 557

“pool\_data\_writes - Buffer pool data writes monitor element” on page 559

“pool\_drty\_pg\_steal\_clns - Buffer pool victim page cleaners triggered monitor element” on page 561

“pool\_drty\_pg\_thrsh\_clns - Buffer pool threshold cleaners triggered monitor element” on page 562

“pool\_index\_l\_reads - Buffer pool index logical reads monitor element” on page 564

“pool\_index\_p\_reads - Buffer pool index physical reads monitor element” on page 566

“pool\_index\_writes - Buffer pool index writes monitor element” on page 567

“pool\_lsn\_gap\_clns - Buffer pool log space cleaners triggered monitor element” on page 569

“pool\_no\_victim\_buffer - Buffer pool no victim buffers monitor element” on page 570

“pool\_read\_time - Total buffer pool physical read time monitor element” on page 571

“pool\_temp\_data\_l\_reads - Buffer pool temporary data logical reads monitor element” on page 573

“pool\_temp\_data\_p\_reads - Buffer pool temporary data physical reads monitor element” on page 575

“pool\_temp\_index\_l\_reads - Buffer pool temporary index logical reads monitor element” on page 577

“pool\_temp\_index\_p\_reads - Buffer pool temporary index physical reads monitor element” on page 578

“pool\_temp\_xda\_l\_reads - Buffer pool temporary XDA data logical reads monitor element” on page 580

“pool\_temp\_xda\_p\_reads - Buffer pool temporary XDA data physical reads monitor element” on page 582

“pool\_write\_time - Total buffer pool physical write time monitor element” on page 584

“pool\_xda\_l\_reads - Buffer pool XDA data logical reads monitor element” on page 585

“pool\_xda\_p\_reads - Buffer pool XDA data physical reads monitor element” on page 588

“pool\_xda\_writes - Buffer pool XDA data writes monitor element” on page 589

“post\_shrthreshold\_hash\_joins - Post threshold hash joins” on page 591

“post\_shrthreshold\_sorts - Post shared threshold sorts monitor element” on page 591

“priv\_workspace\_num\_overflows - Private Workspace Overflows” on page 597

“priv\_workspace\_section\_inserts - Private Workspace Section Inserts” on page 597



"priv\_workspace\_section\_lookups - Private Workspace Section Lookups" on page 598

"priv\_workspace\_size\_top - Maximum Private Workspace Size" on page 599

"rollback\_sql\_stmts - Rollback Statements Attempted" on page 619

"rows\_deleted - Rows deleted monitor element" on page 621

"rows\_inserted - Rows inserted monitor element" on page 622

"rows\_read - Rows read monitor element" on page 624

"rows\_selected - Rows Selected" on page 628

"rows\_updated - Rows updated monitor element" on page 628

"sec\_log\_used\_top - Maximum Secondary Log Space Used" on page 631

"sec\_logs\_allocated - Secondary Logs Allocated Currently" on page 632

"select\_sql\_stmts - Select SQL Statements Executed" on page 634

"server\_platform - Server Operating System" on page 637

"shr\_workspace\_num\_overflows - Shared Workspace Overflows" on page 642

"shr\_workspace\_section\_inserts - Shared Workspace Section Inserts" on page 642

"shr\_workspace\_section\_lookups - Shared Workspace Section Lookups" on page 643

"shr\_workspace\_size\_top - Maximum Shared Workspace Size" on page 643

"sort\_heap\_allocated - Total Sort Heap Allocated" on page 644

"sort\_overflows - Sort overflows monitor element" on page 645

"sort\_shrheap\_allocated - Sort Share Heap Currently Allocated" on page 647

"sort\_shrheap\_top - Sort share heap high watermark" on page 647

"static\_sql\_stmts - Static SQL Statements Attempted" on page 657

"stats\_cache\_size - Size of statistics cache monitor element" on page 658

"stats\_fabricate\_time - Total time spent on statistics fabrication activities monitor element" on page 659

"stats\_fabrications - Total number of statistics fabrications monitor elements" on page 660

"sync\_runstats - Total number of synchronous RUNSTATS activities monitor element" on page 675

"sync\_runstats\_time - Total time spent on synchronous RUNSTATS activities monitor element" on page 676

"tot\_log\_used\_top - Maximum Total Log Space Used" on page 709

"total\_cons - Connects Since Database Activation" on page 711

"total\_hash\_joins - Total Hash Joins" on page 713

"total\_hash\_loops - Total Hash Loops" on page 714

"total\_log\_available - Total Log Available" on page 714

"total\_log\_used - Total Log Space Used" on page 715

"total\_olap\_funcs - Total OLAP Functions monitor element" on page 715

"total\_sec\_cons - Secondary Connections" on page 717

"total\_sort\_time - Total sort time monitor element" on page 720

"total\_sorts - Total sorts monitor element" on page 721

"uid\_sql\_stmts - Update/Insert/Delete SQL Statements Executed" on page 732

"unread\_prefetch\_pages - Unread prefetch pages monitor element" on page 733

"x\_lock\_escals - Exclusive Lock Escalations" on page 748

"xquery\_stmts - XQuery Statements Attempted" on page 750

### **dbase\_remote logical data group**

"commit\_sql\_stmts - Commit Statements Attempted" on page 361  
"create\_nickname - Create Nicknames" on page 387  
"create\_nickname\_time - Create Nickname Response Time" on page 387  
"datasource\_name - Data Source Name" on page 391  
"db\_name - Database Name" on page 393  
"delete\_sql\_stmts - Deletes" on page 401  
"delete\_time - Delete Response Time" on page 402  
"disconnects - Disconnects" on page 414  
"failed\_sql\_stmts - Failed Statement Operations" on page 421  
"insert\_sql\_stmts - Inserts" on page 461  
"insert\_time - Insert Response Time" on page 462  
"passthru\_time - Pass-Through Time" on page 541  
"passthru - Pass-Through" on page 541  
"remote\_lock\_time - Remote Lock Time" on page 611  
"remote\_locks - Remote Locks" on page 611  
"rollback\_sql\_stmts - Rollback Statements Attempted" on page 619  
"rows\_deleted - Rows deleted monitor element" on page 621  
"rows\_inserted - Rows inserted monitor element" on page 622  
"rows\_selected - Rows Selected" on page 628  
"rows\_updated - Rows updated monitor element" on page 628  
"select\_sql\_stmts - Select SQL Statements Executed" on page 634  
"select\_time - Query Response Time" on page 634  
"sp\_rows\_selected - Rows Returned by Stored Procedures" on page 648  
"stored\_proc\_time - Stored Procedure Time" on page 675  
"stored\_procs - Stored Procedures" on page 675  
"total\_cons - Connects Since Database Activation" on page 711  
"update\_sql\_stmts - Updates" on page 738  
"update\_time - Update Response Time" on page 738

### **db\_storage\_group logical data group**

"db\_storage\_path - Automatic storage path monitor element" on page 395  
"fs\_id - Unique file system identification number monitor element" on page 440  
"fs\_total\_size - Total size of a file system monitor element" on page 441  
"fs\_type - File System Type" on page 441  
"fs\_used\_size - Amount of space used on a file system monitor element" on page 442  
"node\_number - Node Number" on page 517  
"sto\_path\_free\_sz - Automatic Storage Path Free Space" on page 674

### **dc\_s\_appl logical data group**

"appl\_idle\_time - Application Idle Time" on page 328  
"commit\_sql\_stmts - Commit Statements Attempted" on page 361  
"elapsed\_exec\_time - Statement Execution Elapsed Time" on page 416

"failed\_sql\_stmts - Failed Statement Operations" on page 421

"gw\_con\_time - DB2 Connect Gateway First Connect Initiated" on page 443

"gw\_exec\_time - Elapsed Time Spent on DB2 Connect Gateway Processing" on page 445

"host\_response\_time - Host Response Time" on page 458

"inbound\_bytes\_received - Inbound Number of Bytes Received" on page 459

"inbound\_bytes\_sent - Inbound Number of Bytes Sent" on page 459

"last\_reset - Last Reset Timestamp" on page 475

"max\_data\_received\_1024 - Number of Statements with Outbound Bytes Received Between 513 and 1024 Bytes" on page 502

"max\_data\_received\_128 - Number of Statements with Outbound Bytes Received Between 1 and 128 Bytes" on page 503

"max\_data\_received\_16384 - Number of Statements with Outbound Bytes Received Between 8193 and 16384 Bytes" on page 503

"max\_data\_received\_2048 - Number of Statements with Outbound Bytes Received Between 1025 and 2048 Bytes" on page 503

"max\_data\_received\_256 - Number of Statements with Outbound Bytes Received Between 129 and 256 Bytes" on page 504

"max\_data\_received\_31999 - Number of Statements with Outbound Bytes Received Between 16385 and 31999 Bytes monitor element" on page 504

"max\_data\_received\_4096 - Number of Statements with Outbound Bytes Received Between 2049 and 4096 Bytes" on page 505

"max\_data\_received\_512 - Number of Statements with Outbound Bytes Received Between 257 and 512 Bytes" on page 505

"max\_data\_received\_64000 - Number of Statements with Outbound Bytes Received Between 32000 and 64000 Bytes monitor element" on page 505

"max\_data\_received\_8192 - Number of Statements with Outbound Bytes Received Between 4097 and 8192 Bytes" on page 506

"max\_data\_received\_gt64000 - Number of Statements with Outbound Bytes Received Greater than 64000 Bytes" on page 506

"max\_data\_sent\_1024 - Number of Statements with Outbound Bytes Sent Between 513 and 1024 Bytes" on page 507

"max\_data\_sent\_128 - Number of Statements with Outbound Bytes Sent Between 1 and 128 Bytes" on page 507

"max\_data\_sent\_16384 - Number of Statements with Outbound Bytes Sent Between 8193 and 16384 Bytes" on page 508

"max\_data\_sent\_2048 - Number of Statements with Outbound Bytes Sent Between 1025 and 2048 Bytes" on page 508

"max\_data\_sent\_256 - Number of Statements with Outbound Bytes Sent Between 129 and 256 Bytes" on page 508

"max\_data\_sent\_31999 - Number of Statements with Outbound Bytes Sent Between 16385 and 31999 Bytes" on page 509

"max\_data\_sent\_4096 - Number of Statements with Outbound Bytes Sent Between 2049 and 4096 Bytes" on page 509

"max\_data\_sent\_512 - Number of Statements with Outbound Bytes Sent Between 257 and 512 Bytes" on page 510

"max\_data\_sent\_64000 - Number of Statements with Outbound Bytes Sent Between 32000 and 64000 Bytes" on page 510

"max\_data\_sent\_8192 - Number of Statements with Outbound Bytes Sent Between 4097 and 8192 Bytes" on page 511  
 "max\_data\_sent\_gt64000 - Number of Statements with Outbound Bytes Sent Greater than 64000 Bytes" on page 511  
 "max\_network\_time\_100\_ms - Number of Statements with Network Time between 16 and 100 ms" on page 512  
 "max\_network\_time\_16\_ms - Number of Statements with Network Time between 4 and 16 ms" on page 512  
 "max\_network\_time\_1\_ms - Number of Statements with Network Time of up to 1 ms" on page 512  
 "max\_network\_time\_4\_ms - Number of Statements with Network Time between 1 and 4 ms" on page 513  
 "max\_network\_time\_500\_ms - Number of Statements with Network Time between 100 and 500 ms" on page 513  
 "max\_network\_time\_gt500\_ms - Number of Statements with Network Time greater than 500 ms" on page 514  
 "network\_time\_bottom - Minimum Network Time for Statement" on page 516  
 "network\_time\_top - Maximum Network Time for Statement" on page 517  
 "open\_cursors - Number of Open Cursors" on page 527  
 "outbound\_bytes\_received - Outbound Number of Bytes Received" on page 530  
 "outbound\_bytes\_sent - Outbound Number of Bytes Sent" on page 531  
 "prev\_uow\_stop\_time - Previous Unit of Work Completion Timestamp" on page 596  
 "rollback\_sql\_stmts - Rollback Statements Attempted" on page 619  
 "rows\_selected - Rows Selected" on page 628  
 "sql\_stmts - Number of SQL Statements Attempted" on page 650  
 "tpmon\_acc\_str - TP monitor client accounting string monitor element" on page 726  
 "tpmon\_client\_app - TP monitor client application name monitor element" on page 726  
 "tpmon\_client\_userid - TP monitor client user ID monitor element" on page 727  
 "tpmon\_client\_wkstn - TP monitor client workstation name monitor element" on page 727  
 "uow\_comp\_status - Unit of Work Completion Status" on page 733  
 "uow\_elapsed\_time - Most Recent Unit of Work Elapsed Time" on page 734  
 "uow\_start\_time - Unit of Work Start Timestamp" on page 736  
 "uow\_stop\_time - Unit of work stop timestamp monitor element" on page 737  
 "xid - Transaction ID" on page 750

### **dc\_s\_appl\_info logical data group**

"agent\_id - Application handle (agent ID) monitor element" on page 313  
 "agent\_status - DCS Application Agents" on page 315  
 "appl\_id - Application ID" on page 325  
 "appl\_name - Application name monitor element" on page 328  
 "auth\_id - Authorization ID" on page 340  
 "client\_pid - Client Process ID" on page 356  
 "client\_platform - Client Operating Platform" on page 357  
 "client\_prdid - Client product and version ID monitor element" on page 357

"client\_protocol - Client Communication Protocol" on page 358  
"codepage\_id - ID of Code Page Used by Application" on page 360  
"dcs\_appl\_status - DCS Application Status" on page 397  
"dcs\_db\_name - DCS Database Name" on page 397  
"execution\_id - User Login ID" on page 421  
"gw\_db\_alias - Database Alias at the Gateway" on page 444  
"host\_ccsid - Host Coded Character Set ID" on page 457  
"host\_db\_name - Host Database Name" on page 457  
"host\_prdid - Host Product/Version ID" on page 457  
"inbound\_comm\_address - Inbound Communication Address" on page 459  
"outbound\_appl\_id - Outbound Application ID" on page 529  
"outbound\_comm\_address - Outbound Communication Address" on page 532  
"outbound\_comm\_protocol - Outbound Communication Protocol" on page 533  
"outbound\_sequence\_no - Outbound Sequence Number" on page 533  
"sequence\_no - Sequence number monitor element" on page 635  
"status\_change\_time - Application Status Change Time" on page 660

### **dcs\_dbase logical data group**

"commit\_sql\_stmts - Commit Statements Attempted" on page 361  
"con\_elapsed\_time - Most Recent Connection Elapsed Time" on page 362  
"con\_response\_time - Most Recent Response Time for Connect" on page 363  
"dcs\_db\_name - DCS Database Name" on page 397  
"elapsed\_exec\_time - Statement Execution Elapsed Time" on page 416  
"failed\_sql\_stmts - Failed Statement Operations" on page 421  
"gw\_comm\_error\_time - Communication Error Time" on page 442  
"gw\_comm\_errors - Communication Errors" on page 442  
"gw\_con\_time - DB2 Connect Gateway First Connect Initiated" on page 443  
"gw\_connections\_top - Maximum Number of Concurrent Connections to Host Database" on page 443  
"gw\_cons\_wait\_client - Number of Connections Waiting for the Client to Send Request" on page 443  
"gw\_cons\_wait\_host - Number of Connections Waiting for the Host to Reply" on page 444  
"gw\_cur\_cons - Current Number of Connections for DB2 Connect" on page 444  
"gw\_total\_cons - Total Number of Attempted Connections for DB2 Connect" on page 445  
"host\_db\_name - Host Database Name" on page 457  
"host\_response\_time - Host Response Time" on page 458  
"inbound\_bytes\_received - Inbound Number of Bytes Received" on page 459  
"last\_reset - Last Reset Timestamp" on page 475  
"max\_data\_received\_1024 - Number of Statements with Outbound Bytes Received Between 513 and 1024 Bytes" on page 502  
"max\_data\_received\_128 - Number of Statements with Outbound Bytes Received Between 1 and 128 Bytes" on page 503  
"max\_data\_received\_16384 - Number of Statements with Outbound Bytes Received Between 8193 and 16384 Bytes" on page 503

“max\_data\_received\_2048 - Number of Statements with Outbound Bytes Received Between 1025 and 2048 Bytes” on page 503

“max\_data\_received\_256 - Number of Statements with Outbound Bytes Received Between 129 and 256 Bytes” on page 504

“max\_data\_received\_31999 - Number of Statements with Outbound Bytes Received Between 16385 and 31999 Bytes monitor element” on page 504

“max\_data\_received\_4096 - Number of Statements with Outbound Bytes Received Between 2049 and 4096 Bytes” on page 505

“max\_data\_received\_512 - Number of Statements with Outbound Bytes Received Between 257 and 512 Bytes” on page 505

“max\_data\_received\_64000 - Number of Statements with Outbound Bytes Received Between 32000 and 64000 Bytes monitor element” on page 505

“max\_data\_received\_8192 - Number of Statements with Outbound Bytes Received Between 4097 and 8192 Bytes” on page 506

“max\_data\_received\_gt64000 - Number of Statements with Outbound Bytes Received Greater than 64000 Bytes” on page 506

“max\_data\_sent\_1024 - Number of Statements with Outbound Bytes Sent Between 513 and 1024 Bytes” on page 507

“max\_data\_sent\_128 - Number of Statements with Outbound Bytes Sent Between 1 and 128 Bytes” on page 507

“max\_data\_sent\_16384 - Number of Statements with Outbound Bytes Sent Between 8193 and 16384 Bytes” on page 508

“max\_data\_sent\_2048 - Number of Statements with Outbound Bytes Sent Between 1025 and 2048 Bytes” on page 508

“max\_data\_sent\_256 - Number of Statements with Outbound Bytes Sent Between 129 and 256 Bytes” on page 508

“max\_data\_sent\_31999 - Number of Statements with Outbound Bytes Sent Between 16385 and 31999 Bytes” on page 509

“max\_data\_sent\_4096 - Number of Statements with Outbound Bytes Sent Between 2049 and 4096 Bytes” on page 509

“max\_data\_sent\_512 - Number of Statements with Outbound Bytes Sent Between 257 and 512 Bytes” on page 510

“max\_data\_sent\_64000 - Number of Statements with Outbound Bytes Sent Between 32000 and 64000 Bytes” on page 510

“max\_data\_sent\_8192 - Number of Statements with Outbound Bytes Sent Between 4097 and 8192 Bytes” on page 511

“max\_data\_sent\_gt64000 - Number of Statements with Outbound Bytes Sent Greater than 64000 Bytes” on page 511

“max\_network\_time\_100\_ms - Number of Statements with Network Time between 16 and 100 ms” on page 512

“max\_network\_time\_16\_ms - Number of Statements with Network Time between 4 and 16 ms” on page 512

“max\_network\_time\_1\_ms - Number of Statements with Network Time of up to 1 ms” on page 512

“max\_network\_time\_4\_ms - Number of Statements with Network Time between 1 and 4 ms” on page 513

“max\_network\_time\_500\_ms - Number of Statements with Network Time between 100 and 500 ms” on page 513

“max\_network\_time\_gt500\_ms - Number of Statements with Network Time greater than 500 ms” on page 514



"network\_time\_bottom - Minimum Network Time for Statement" on page 516  
"network\_time\_top - Maximum Network Time for Statement" on page 517  
"outbound\_bytes\_sent - Outbound Number of Bytes Sent" on page 531  
"rollback\_sql\_stmts - Rollback Statements Attempted" on page 619  
"rows\_selected - Rows Selected" on page 628  
"sql\_stmts - Number of SQL Statements Attempted" on page 650

### **dcs\_stmt logical data group**

"blocking\_cursor - Blocking Cursor" on page 345  
"creator - Application Creator" on page 387  
"elapsed\_exec\_time - Statement Execution Elapsed Time" on page 416  
"fetch\_count - Number of Successful Fetches" on page 438  
"gw\_exec\_time - Elapsed Time Spent on DB2 Connect Gateway Processing" on page 445  
"host\_response\_time - Host Response Time" on page 458  
"inbound\_bytes\_received - Inbound Number of Bytes Received" on page 459  
"inbound\_bytes\_sent - Inbound Number of Bytes Sent" on page 459  
"num\_transmissions - Number of Transmissions" on page 525  
"num\_transmissions\_group - Number of Transmissions Group" on page 526  
"outbound\_bytes\_received - Outbound Number of Bytes Received" on page 530  
"outbound\_bytes\_sent - Outbound Number of Bytes Sent" on page 531  
"package\_name - Package name monitor element" on page 534  
"query\_card\_estimate - Query Number of Rows Estimate" on page 603  
"query\_cost\_estimate - Query cost estimate monitor element" on page 604  
"section\_number - Section number monitor element" on page 633  
"stmt\_elapsed\_time - Most Recent Statement Elapsed Time" on page 660  
"stmt\_operation/operation - Statement operation monitor element" on page 664  
"stmt\_start - Statement Operation Start Timestamp" on page 668  
"stmt\_stop - Statement Operation Stop Timestamp" on page 668  
"stmt\_text - SQL statement text monitor element" on page 669

### **detail\_log logical data group**

"current\_active\_log - Current Active Log File Number" on page 388  
"current\_archive\_log - Current Archive Log File Number" on page 388  
"first\_active\_log - First Active Log File Number" on page 439  
"last\_active\_log - Last Active Log File Number" on page 473  
"node\_number - Node Number" on page 517

### **dynsql logical data group**

"fetch\_count - Number of Successful Fetches" on page 438  
"insert\_timestamp - Statement insert timestamp monitor element" on page 462  
"int\_rows\_deleted - Internal Rows Deleted" on page 466  
"int\_rows\_inserted - Internal Rows Inserted" on page 466  
"int\_rows\_updated - Internal Rows Updated" on page 467  
"num\_compilations - Statement Compilations" on page 519  
"num\_executions - Statement executions monitor element" on page 520

"pool\_data\_l\_reads - Buffer pool data logical reads monitor element" on page 556

"pool\_data\_p\_reads - Buffer pool data physical reads monitor element" on page 557

"pool\_index\_l\_reads - Buffer pool index logical reads monitor element" on page 564

"pool\_index\_p\_reads - Buffer pool index physical reads monitor element" on page 566

"pool\_temp\_data\_l\_reads - Buffer pool temporary data logical reads monitor element" on page 573

"pool\_temp\_data\_p\_reads - Buffer pool temporary data physical reads monitor element" on page 575

"pool\_temp\_index\_l\_reads - Buffer pool temporary index logical reads monitor element" on page 577

"pool\_temp\_index\_p\_reads - Buffer pool temporary index physical reads monitor element" on page 578

"pool\_temp\_xda\_l\_reads - Buffer pool temporary XDA data logical reads monitor element" on page 580

"pool\_temp\_xda\_p\_reads - Buffer pool temporary XDA data physical reads monitor element" on page 582

"pool\_xda\_l\_reads - Buffer pool XDA data logical reads monitor element" on page 585

"pool\_xda\_p\_reads - Buffer pool XDA data physical reads monitor element" on page 588

"prep\_time\_best - Statement best preparation time monitor element" on page 596

"prep\_time\_worst - Statement worst preparation time monitor element" on page 596

"rows\_read - Rows read monitor element" on page 624

"rows\_written - Rows Written" on page 629

"sort\_overflows - Sort overflows monitor element" on page 645

"stats\_fabricate\_time - Total time spent on statistics fabrication activities monitor element" on page 659

"stmt\_pkgcache\_id - Statement package cache identifier" on page 666

"stmt\_sorts - Statement Sorts" on page 667

"stmt\_text - SQL statement text monitor element" on page 669

"sync\_runstats\_time - Total time spent on synchronous RUNSTATS activities monitor element" on page 676

"total\_exec\_time - Elapsed Statement Execution Time" on page 713

"total\_sort\_time - Total sort time monitor element" on page 720

"total\_sys\_cpu\_time - Total system CPU time for a statement monitor element" on page 723

"total\_usr\_cpu\_time - Total user CPU time for a statement monitor element" on page 725

### **dynsql\_list logical data group**

"db\_name - Database Name" on page 393

"db\_path - Database Path" on page 394



### **fcml logical data group**

- “buff\_free - FCM Buffers Currently Free” on page 348
- “buff\_free\_bottom - Minimum FCM Buffers Free” on page 348
- “ch\_free - Channels Currently Free” on page 353
- “ch\_free\_bottom - Minimum Channels Free” on page 353

### **fcml\_node logical data group**

- “connection\_status - Connection Status” on page 372
- “node\_number - Node Number” on page 517
- “total\_buffers\_rcvd - Total FCM Buffers Received” on page 711
- “total\_buffers\_sent - Total FCM Buffers Sent” on page 711

### **hadr logical data group**

- “hadr\_connect\_status - HADR Connection Status monitor element” on page 445
- “hadr\_connect\_time - HADR Connection Time monitor element” on page 446
- “hadr\_heartbeat - HADR Heartbeat monitor element” on page 447
- “hadr\_local\_host - HADR Local Host monitor element” on page 447
- “hadr\_local\_service - HADR Local Service monitor element” on page 448
- “hadr\_log\_gap - HADR Log Gap” on page 448
- “hadr\_primary\_log\_file - HADR Primary Log File monitor element” on page 449
- “hadr\_primary\_log\_lsn - HADR Primary Log LSN monitor element” on page 450
- “hadr\_primary\_log\_page - HADR Primary Log Page monitor element” on page 450
- “hadr\_remote\_host - HADR Remote Host monitor element” on page 450
- “hadr\_remote\_instance - HADR Remote Instance monitor element” on page 451
- “hadr\_remote\_service - HADR Remote Service monitor element” on page 451
- “hadr\_role - HADR Role” on page 452
- “hadr\_standby\_log\_file - HADR Standby Log File monitor element” on page 452
- “hadr\_standby\_log\_lsn - HADR Standby Log LSN monitor element” on page 452
- “hadr\_standby\_log\_page - HADR Standby Log Page monitor element” on page 453
- “hadr\_state - HADR State monitor element” on page 453
- “hadr\_syncmode - HADR Synchronization Mode monitor element” on page 454
- “hadr\_timeout - HADR Timeout monitor element” on page 454

### **lock logical data group**

- “data\_partition\_id - Data partition identifier monitor element” on page 390
- “lock\_attributes - Lock attributes monitor element” on page 477
- “lock\_count - Lock count monitor element” on page 478
- “lock\_current\_mode - Original Lock Mode Before Conversion” on page 479
- “lock\_escalation - Lock escalation monitor element” on page 480
- “lock\_hold\_count - Lock hold count monitor element” on page 482
- “lock\_mode - Lock mode monitor element” on page 483

“lock\_name - Lock name monitor element” on page 484  
“lock\_object\_name - Lock Object Name” on page 485  
“lock\_object\_type - Lock object type waited on monitor element” on page 486  
“lock\_release\_flags - Lock release flags monitor element” on page 486  
“lock\_status - Lock status monitor element” on page 487  
“node\_number - Node Number” on page 517  
“table\_file\_id - Table file ID monitor element” on page 678  
“table\_name - Table name monitor element” on page 678  
“table\_schema - Table schema name monitor element” on page 679  
“tablespace\_name - Table space name monitor element” on page 687

### **lock\_wait logical data group**

“agent\_id\_holding\_lock - Agent ID Holding Lock” on page 314  
“appl\_id\_holding\_lk - Application ID Holding Lock” on page 327  
“data\_partition\_id - Data partition identifier monitor element” on page 390  
“lock\_attributes - Lock attributes monitor element” on page 477  
“lock\_current\_mode - Original Lock Mode Before Conversion” on page 479  
“lock\_escalation - Lock escalation monitor element” on page 480  
“lock\_mode - Lock mode monitor element” on page 483  
“lock\_mode\_requested - Lock mode requested monitor element” on page 484  
“lock\_name - Lock name monitor element” on page 484  
“lock\_object\_type - Lock object type waited on monitor element” on page 486  
“lock\_release\_flags - Lock release flags monitor element” on page 486  
“lock\_wait\_start\_time - Lock Wait Start Timestamp” on page 490  
“node\_number - Node Number” on page 517  
“ss\_number - Subsection Number” on page 655  
“table\_name - Table name monitor element” on page 678  
“table\_schema - Table schema name monitor element” on page 679  
“tablespace\_name - Table space name monitor element” on page 687

### **memory\_pool logical data group**

“node\_number - Node Number” on page 517  
“pool\_config\_size - Configured Size of Memory Pool” on page 554  
“pool\_cur\_size - Current Size of Memory Pool” on page 555  
“pool\_id - Memory Pool Identifier” on page 563  
“pool\_secondary\_id - Memory Pool Secondary Identifier” on page 573  
“pool\_watermark - Memory Pool Watermark” on page 583

### **progress logical data group**

“progress\_completed\_units - Completed Progress Work Units” on page 599  
“progress\_description - Progress Description” on page 600  
“progress\_seq\_num - Progress Sequence Number” on page 601  
“progress\_start\_time - Progress Start Time” on page 601  
“progress\_total\_units - Total Progress Work Units” on page 601  
“progress\_work\_metric - Progress Work Metric” on page 602

## **progress\_list logical data group**

“progress\_list\_attr - Current Progress List Attributes” on page 600

“progress\_list\_cur\_seq\_num - Current Progress List Sequence Number” on page 601

## **rollforward logical data group**

“node\_number - Node Number” on page 517

“rf\_log\_num - Log Being Rolled Forward” on page 617

“rf\_status - Log Phase” on page 618

“rf\_timestamp - Rollforward Timestamp” on page 618

“rf\_type - Rollforward Type” on page 618

“ts\_name - Table space being rolled forward monitor element” on page 732

## **stmt logical data group**

“agents\_top - Number of Agents Created” on page 321

“blocking\_cursor - Blocking Cursor” on page 345

“consistency\_token - Package consistency token monitor element” on page 373

“creator - Application Creator” on page 387

“cursor\_name - Cursor Name” on page 389

“degree\_parallelism - Degree of Parallelism” on page 401

“fetch\_count - Number of Successful Fetches” on page 438

“int\_rows\_deleted - Internal Rows Deleted” on page 466

“int\_rows\_inserted - Internal Rows Inserted” on page 466

“int\_rows\_updated - Internal Rows Updated” on page 467

“num\_agents - Number of Agents Working on a Statement” on page 518

“package\_name - Package name monitor element” on page 534

“package\_version\_id - Package version monitor element” on page 535

“pool\_data\_l\_reads - Buffer pool data logical reads monitor element” on page 556

“pool\_data\_p\_reads - Buffer pool data physical reads monitor element” on page 557

“pool\_index\_l\_reads - Buffer pool index logical reads monitor element” on page 564

“pool\_index\_p\_reads - Buffer pool index physical reads monitor element” on page 566

“pool\_temp\_data\_l\_reads - Buffer pool temporary data logical reads monitor element” on page 573

“pool\_temp\_data\_p\_reads - Buffer pool temporary data physical reads monitor element” on page 575

“pool\_temp\_index\_l\_reads - Buffer pool temporary index logical reads monitor element” on page 577

“pool\_temp\_index\_p\_reads - Buffer pool temporary index physical reads monitor element” on page 578

“pool\_temp\_xda\_l\_reads - Buffer pool temporary XDA data logical reads monitor element” on page 580

“pool\_temp\_xda\_p\_reads - Buffer pool temporary XDA data physical reads monitor element” on page 582

"pool\_xda\_l\_reads - Buffer pool XDA data logical reads monitor element" on page 585  
 "pool\_xda\_p\_reads - Buffer pool XDA data physical reads monitor element" on page 588  
 "query\_card\_estimate - Query Number of Rows Estimate" on page 603  
 "query\_cost\_estimate - Query cost estimate monitor element" on page 604  
 "rows\_read - Rows read monitor element" on page 624  
 "rows\_written - Rows Written" on page 629  
 "section\_number - Section number monitor element" on page 633  
 "sort\_overflows - Sort overflows monitor element" on page 645  
 "stmt\_elapsed\_time - Most Recent Statement Elapsed Time" on page 660  
 "stmt\_node\_number - Statement Node" on page 664  
 "stmt\_operation/operation - Statement operation monitor element" on page 664  
 "stmt\_sorts - Statement Sorts" on page 667  
 "stmt\_start - Statement Operation Start Timestamp" on page 668  
 "stmt\_stop - Statement Operation Stop Timestamp" on page 668  
 "stmt\_sys\_cpu\_time - System CPU Time used by Statement" on page 669  
 "stmt\_text - SQL statement text monitor element" on page 669  
 "stmt\_type - Statement type monitor element" on page 670  
 "stmt\_usr\_cpu\_time - User CPU Time used by Statement" on page 671  
 "total\_sort\_time - Total sort time monitor element" on page 720

### **stmt\_transmissions logical data group**

"elapsed\_exec\_time - Statement Execution Elapsed Time" on page 416  
 "host\_response\_time - Host Response Time" on page 458  
 "max\_data\_received\_1024 - Number of Statements with Outbound Bytes Received Between 513 and 1024 Bytes" on page 502  
 "max\_data\_received\_128 - Number of Statements with Outbound Bytes Received Between 1 and 128 Bytes" on page 503  
 "max\_data\_received\_16384 - Number of Statements with Outbound Bytes Received Between 8193 and 16384 Bytes" on page 503  
 "max\_data\_received\_2048 - Number of Statements with Outbound Bytes Received Between 1025 and 2048 Bytes" on page 503  
 "max\_data\_received\_256 - Number of Statements with Outbound Bytes Received Between 129 and 256 Bytes" on page 504  
 "max\_data\_received\_31999 - Number of Statements with Outbound Bytes Received Between 16385 and 31999 Bytes monitor element" on page 504  
 "max\_data\_received\_4096 - Number of Statements with Outbound Bytes Received Between 2049 and 4096 Bytes" on page 505  
 "max\_data\_received\_512 - Number of Statements with Outbound Bytes Received Between 257 and 512 Bytes" on page 505  
 "max\_data\_received\_64000 - Number of Statements with Outbound Bytes Received Between 32000 and 64000 Bytes monitor element" on page 505  
 "max\_data\_received\_8192 - Number of Statements with Outbound Bytes Received Between 4097 and 8192 Bytes" on page 506  
 "max\_data\_received\_gt64000 - Number of Statements with Outbound Bytes Received Greater than 64000 Bytes" on page 506

"max\_data\_sent\_1024 - Number of Statements with Outbound Bytes Sent Between 513 and 1024 Bytes" on page 507

"max\_data\_sent\_128 - Number of Statements with Outbound Bytes Sent Between 1 and 128 Bytes" on page 507

"max\_data\_sent\_16384 - Number of Statements with Outbound Bytes Sent Between 8193 and 16384 Bytes" on page 508

"max\_data\_sent\_2048 - Number of Statements with Outbound Bytes Sent Between 1025 and 2048 Bytes" on page 508

"max\_data\_sent\_256 - Number of Statements with Outbound Bytes Sent Between 129 and 256 Bytes" on page 508

"max\_data\_sent\_31999 - Number of Statements with Outbound Bytes Sent Between 16385 and 31999 Bytes" on page 509

"max\_data\_sent\_4096 - Number of Statements with Outbound Bytes Sent Between 2049 and 4096 Bytes" on page 509

"max\_data\_sent\_512 - Number of Statements with Outbound Bytes Sent Between 257 and 512 Bytes" on page 510

"max\_data\_sent\_64000 - Number of Statements with Outbound Bytes Sent Between 32000 and 64000 Bytes" on page 510

"max\_data\_sent\_8192 - Number of Statements with Outbound Bytes Sent Between 4097 and 8192 Bytes" on page 511

"max\_data\_sent\_gt64000 - Number of Statements with Outbound Bytes Sent Greater than 64000 Bytes" on page 511

"max\_network\_time\_100\_ms - Number of Statements with Network Time between 16 and 100 ms" on page 512

"max\_network\_time\_16\_ms - Number of Statements with Network Time between 4 and 16 ms" on page 512

"max\_network\_time\_1\_ms - Number of Statements with Network Time of up to 1 ms" on page 512

"max\_network\_time\_4\_ms - Number of Statements with Network Time between 1 and 4 ms" on page 513

"max\_network\_time\_500\_ms - Number of Statements with Network Time between 100 and 500 ms" on page 513

"max\_network\_time\_gt500\_ms - Number of Statements with Network Time greater than 500 ms" on page 514

"network\_time\_bottom - Minimum Network Time for Statement" on page 516

"network\_time\_top - Maximum Network Time for Statement" on page 517

"outbound\_bytes\_received - Outbound Number of Bytes Received" on page 530

"outbound\_bytes\_received\_bottom - Minimum Outbound Number of Bytes Received" on page 531

"outbound\_bytes\_received\_top - Maximum Outbound Number of Bytes Received" on page 531

"outbound\_bytes\_sent - Outbound Number of Bytes Sent" on page 531

"outbound\_bytes\_sent\_bottom - Minimum Outbound Number of Bytes Sent" on page 532

"outbound\_bytes\_sent\_top - Maximum Outbound Number of Bytes Sent" on page 532

"sql\_chains - Number of SQL Chains Attempted" on page 649

"sql\_stmts - Number of SQL Statements Attempted" on page 650

## **subsection logical data group**

- "rows\_read - Rows read monitor element" on page 624*
- "rows\_written - Rows Written" on page 629*
- "ss\_exec\_time - Subsection Execution Elapsed Time" on page 655*
- "ss\_node\_number - Subsection Node Number" on page 655*
- "ss\_number - Subsection Number" on page 655*
- "ss\_status - Subsection Status" on page 656*
- "ss\_sys\_cpu\_time - System CPU Time used by Subsection" on page 656*
- "ss\_usr\_cpu\_time - User CPU Time used by Subsection" on page 656*
- "tq\_cur\_send\_spills - Current number of table queue buffers overflowed monitor element" on page 728*
- "tq\_id\_waiting\_on - Waited on node on a table queue" on page 728*
- "tq\_max\_send\_spills - Maximum number of table queue buffers overflows" on page 729*
- "tq\_node\_waited\_for - Waited for node on a table queue" on page 729*
- "tq\_rows\_read - Number of Rows Read from table queues" on page 729*
- "tq\_rows\_written - Number of rows written to table queues" on page 730*
- "tq\_tot\_send\_spills - Total number of table queue buffers overflowed monitor element" on page 730*
- "tq\_wait\_for\_any - Waiting for any node to send on a table queue" on page 731*

## **table logical data group**

- "data\_object\_pages - Data Object Pages" on page 390*
- "data\_partition\_id - Data partition identifier monitor element" on page 390*
- "index\_object\_pages - Index Object Pages" on page 460*
- "lob\_object\_pages - LOB Object Pages" on page 476*
- "long\_object\_pages - Long Object Pages" on page 501*
- "overflow\_accesses - Accesses to overflowed records monitor element" on page 533*
- "page\_reorgs - Page Reorganizations" on page 536*
- "rows\_read - Rows read monitor element" on page 624*
- "rows\_written - Rows Written" on page 629*
- "table\_file\_id - Table file ID monitor element" on page 678*
- "table\_name - Table name monitor element" on page 678*
- "table\_schema - Table schema name monitor element" on page 679*
- "table\_type - Table type monitor element" on page 681*
- "tablespace\_id - Table space identification monitor element" on page 684*
- "xda\_object\_pages - XDA Object Pages" on page 749*

## **table\_list logical data group**

- "db\_conn\_time - Database activation timestamp monitor element" on page 392*
- "db\_name - Database Name" on page 393*
- "db\_path - Database Path" on page 394*
- "input\_db\_alias - Input Database Alias" on page 461*
- "last\_reset - Last Reset Timestamp" on page 475*

## **table\_reorg logical data group**

- “data\_partition\_id - Data partition identifier monitor element” on page 390
- “reorg\_completion - Reorganization Completion Flag” on page 611
- “reorg\_current\_counter - Reorganize Progress” on page 612
- “reorg\_end - Table Reorganize End Time” on page 612
- “reorg\_index\_id - Index Used to Reorganize the Table” on page 612
- “reorg\_max\_counter - Total Amount of Reorganization” on page 613
- “reorg\_max\_phase - Maximum Reorganize Phase” on page 613
- “reorg\_phase - Table reorganization phase monitor element” on page 613
- “reorg\_phase\_start - Reorganize Phase Start Time” on page 614
- “reorg\_rows\_compressed - Rows Compressed” on page 614
- “reorg\_rows\_rejected\_for\_compression - Rows Rejected for Compression” on page 614
- “reorg\_start - Table Reorganize Start Time” on page 615
- “reorg\_status - Table Reorganize Status” on page 615
- “reorg\_tbsp\_id - Table Space Where Table or Data partition is Reorganized” on page 615
- “reorg\_type - Table Reorganize Attributes” on page 616
- “reorg\_xml\_regions\_compressed - XML regions compressed monitor element” on page 616
- “reorg\_xml\_regions\_rejected\_for\_compression - XML regions rejected for compression monitor element” on page 617

## **tablespace logical data group**

- “direct\_read\_reqs - Direct read requests monitor element” on page 404
- “direct\_read\_time - Direct read time monitor element” on page 406
- “direct\_reads - Direct reads from database monitor element” on page 407
- “direct\_write\_reqs - Direct write requests monitor element” on page 409
- “direct\_write\_time - Direct write time monitor element” on page 410
- “direct\_writes - Direct writes to database monitor element” on page 412
- “files\_closed - Database files closed monitor element” on page 438
- “fs\_caching - File system caching monitor element” on page 440
- “pool\_async\_data\_read\_reqs - Buffer pool asynchronous read requests monitor element” on page 546
- “pool\_async\_data\_reads - Buffer pool asynchronous data reads monitor element” on page 547
- “pool\_async\_data\_writes - Buffer pool asynchronous data writes monitor element” on page 547
- “pool\_async\_index\_read\_reqs - Buffer pool asynchronous index read requests monitor element” on page 548
- “pool\_async\_index\_reads - Buffer pool asynchronous index reads monitor element” on page 549
- “pool\_async\_index\_writes - Buffer pool asynchronous index writes monitor element” on page 550
- “pool\_async\_read\_time - Buffer Pool Asynchronous Read Time” on page 551
- “pool\_async\_write\_time - Buffer Pool Asynchronous Write Time” on page 551



“pool\_async\_xda\_read\_reqs - Buffer pool asynchronous XDA read requests monitor element” on page 552

“pool\_async\_xda\_reads - Buffer pool asynchronous XDA data reads monitor element” on page 553

“pool\_async\_xda\_writes - Buffer pool asynchronous XDA data writes monitor element” on page 554

“pool\_data\_l\_reads - Buffer pool data logical reads monitor element” on page 556

“pool\_data\_p\_reads - Buffer pool data physical reads monitor element” on page 557

“pool\_data\_writes - Buffer pool data writes monitor element” on page 559

“pool\_index\_l\_reads - Buffer pool index logical reads monitor element” on page 564

“pool\_index\_p\_reads - Buffer pool index physical reads monitor element” on page 566

“pool\_index\_writes - Buffer pool index writes monitor element” on page 567

“pool\_no\_victim\_buffer - Buffer pool no victim buffers monitor element” on page 570

“pool\_read\_time - Total buffer pool physical read time monitor element” on page 571

“pool\_temp\_data\_l\_reads - Buffer pool temporary data logical reads monitor element” on page 573

“pool\_temp\_data\_p\_reads - Buffer pool temporary data physical reads monitor element” on page 575

“pool\_temp\_index\_l\_reads - Buffer pool temporary index logical reads monitor element” on page 577

“pool\_temp\_index\_p\_reads - Buffer pool temporary index physical reads monitor element” on page 578

“pool\_temp\_xda\_l\_reads - Buffer pool temporary XDA data logical reads monitor element” on page 580

“pool\_temp\_xda\_p\_reads - Buffer pool temporary XDA data physical reads monitor element” on page 582

“pool\_write\_time - Total buffer pool physical write time monitor element” on page 584

“pool\_xda\_l\_reads - Buffer pool XDA data logical reads monitor element” on page 585

“pool\_xda\_p\_reads - Buffer pool XDA data physical reads monitor element” on page 588

“pool\_xda\_writes - Buffer pool XDA data writes monitor element” on page 589

“tablespace\_auto\_resize\_enabled - Table space automatic resizing enabled monitor element” on page 681

“tablespace\_content\_type - Table space content type monitor element” on page 682

“tablespace\_cur\_pool\_id - Buffer pool currently being used monitor element” on page 682

“tablespace\_extent\_size - Table space extent size monitor element” on page 683

“tablespace\_id - Table space identification monitor element” on page 684

“tablespace\_name - Table space name monitor element” on page 687



*"tablespace\_next\_pool\_id - Buffer pool that will be used at next startup monitor element" on page 688*

*"tablespace\_page\_size - Table space page size monitor element" on page 689*

*"tablespace\_prefetch\_size - Table space prefetch size monitor element" on page 690*

*"tablespace\_rebalancer\_mode - Rebalancer mode monitor element" on page 692*

*"tablespace\_type - Table space type monitor element" on page 696*

*"tablespace\_using\_auto\_storage - Table space enabled for automatic storage monitor element" on page 698*

### **tablespace\_container logical data group**

*"container\_accessible - Accessibility of container monitor element" on page 373*

*"container\_id - Container identification monitor element" on page 373*

*"container\_name - Container name monitor element" on page 374*

*"container\_stripe\_set - Container stripe set monitor element" on page 374*

*"container\_total\_pages - Total pages in container monitor element" on page 375*

*"container\_type - Container type monitor element" on page 375*

*"container\_usable\_pages - Usable pages in container monitor element" on page 375*

### **tablespace\_list logical data group**

*"db\_conn\_time - Database activation timestamp monitor element" on page 392*

*"db\_name - Database Name" on page 393*

*"db\_path - Database Path" on page 394*

*"input\_db\_alias - Input Database Alias" on page 461*

*"last\_reset - Last Reset Timestamp" on page 475*

### **tablespace\_nodeinfo logical data group**

*"tablespace\_current\_size - Current table space size" on page 683*

*"tablespace\_free\_pages - Free pages in table space monitor element" on page 683*

*"tablespace\_increase\_size - Increase size in bytes" on page 684*

*"tablespace\_increase\_size\_percent - Increase size by percent monitor element" on page 685*

*"tablespace\_initial\_size - Initial table space size" on page 685*

*"tablespace\_last\_resize\_failed - Last resize attempt failed" on page 685*

*"tablespace\_last\_resize\_time - Time of last successful resize" on page 686*

*"tablespace\_max\_size - Maximum table space size" on page 686*

*"tablespace\_min\_recovery\_time - Minimum Recovery Time For Rollforward" on page 686*

*"tablespace\_num\_containers - Number of Containers in Table Space" on page 688*

*"tablespace\_num\_quiescers - Number of Quiescers" on page 688*

*"tablespace\_num\_ranges - Number of Ranges in the Table Space Map" on page 689*

*"tablespace\_page\_top - Table space high watermark monitor element" on page 689*

*“tablespace\_paths\_dropped - Table space using dropped path monitor element” on page 689*

*“tablespace\_pending\_free\_pages - Pending free pages in table space monitor element” on page 690*

*“tablespace\_prefetch\_size - Table space prefetch size monitor element” on page 690*

*“tablespace\_rebalancer\_extents\_processed - Number of Extents the Rebalancer has Processed” on page 691*

*“tablespace\_rebalancer\_extents\_remaining - Total Number of Extents to be Processed by the Rebalancer” on page 691*

*“tablespace\_rebalancer\_last\_extent\_moved - Last Extent Moved by the Rebalancer” on page 692*

*“tablespace\_rebalancer\_priority - Current Rebalancer Priority” on page 693*

*“tablespace\_rebalancer\_restart\_time - Rebalancer Restart Time” on page 693*

*“tablespace\_rebalancer\_start\_time - Rebalancer Start Time” on page 694*

*“tablespace\_state - Table space state monitor element” on page 694*

*“tablespace\_state\_change\_object\_id - State Change Object Identification” on page 695*

*“tablespace\_state\_change\_ts\_id - State Change Table Space Identification” on page 695*

*“tablespace\_total\_pages - Total pages in table space monitor element” on page 696*

*“tablespace\_usable\_pages - Usable pages in table space monitor element” on page 697*

*“tablespace\_used\_pages - Used pages in table space monitor element” on page 697*

### **tablespace\_quiescer logical data group**

*“quiescer\_agent\_id - Quiescer Agent Identification” on page 605*

*“quiescer\_auth\_id - Quiescer User Authorization Identification” on page 605*

*“quiescer\_obj\_id - Quiescer Object Identification” on page 606*

*“quiescer\_state - Quiescer State” on page 606*

*“quiescer\_ts\_id - Quiescer Table Space Identification” on page 606*

### **tablespace\_range logical data group**

*“range\_adjustment - Range Adjustment” on page 607*

*“range\_container\_id - Range Container” on page 607*

*“range\_end\_stripe - End Stripe” on page 607*

*“range\_max\_extent - Maximum Extent in Range” on page 607*

*“range\_max\_page\_number - Maximum Page in Range” on page 608*

*“range\_num\_containers - Number of Containers in Range” on page 608*

*“range\_number - Range Number” on page 608*

*“range\_offset - Range Offset” on page 608*

*“range\_start\_stripe - Start Stripe” on page 608*

*“range\_stripe\_set\_number - Stripe Set Number” on page 609*

## utility\_info logical data group

- “node\_number - Node Number” on page 517
- “utility\_dbname - Database Operated on by Utility” on page 739
- “utility\_description - Utility Description” on page 739
- “utility\_id - Utility ID” on page 740
- “utility\_invoker\_type - Utility Invoker Type” on page 740
- “utility\_priority - Utility Priority” on page 740
- “utility\_start\_time - Utility Start Time” on page 740
- “utility\_state - Utility State” on page 741
- “utility\_type - Utility Type” on page 741

---

## Event type mappings to logical data groups

For file, pipe and table event monitors, event monitor output consists of an ordered series of logical data groupings. Regardless of the event monitor type, the output records always contain the same starting logical data groups. These frame the logical data groups whose presence varies depending on the event types recorded by the event monitor.

For file and pipe event monitors, event records may be generated for any connection and may therefore appear in mixed order in the stream. This means that you may get a transaction event for Connection 1, immediately followed by a connection event for Connection 2. However, records belonging to a single connection or a single event will appear in their logical order. For example, a statement record (end of statement) always precedes a transaction record (end of UOW), if any. Similarly, a deadlock event record always precedes the deadlocked connection event records for each connection involved in the deadlock. The **application id** or **application handle (agent\_id)** can be used to match records with a connection.

Connection header events are normally written for each connection to the database. For deadlocks with details event monitors, they are only written when the deadlock occurs. In this case, connection header events are only written for participants in the deadlock and not for all connections to the database.

The logical data groupings are ordered according to four different levels: Monitor, Prolog, Contents, and Epilog. Following are detailed descriptions for each level, including the corresponding event types and logical data groups.

### Monitor

Information at the Monitor level is generated for all event monitors. It consists of event monitor metadata.

*Table 72. Event Monitor Data Stream: Monitor Section*

Event type	Logical data group	Available information
Monitor Level	event_log_stream_header	Identifies the version level and byte order of the event monitor. Applications can use this header to determine whether they can handle the evmon output stream.

## Prolog

The Prolog information is generated when the event monitor is activated.

*Table 73. Event Monitor Data Stream: Prolog Section*

Event type	Logical data group	Available information
Log Header	event_log_header	Characteristics of the trace, for example server type and memory layout.
Database Header	event_db_header	Database name, path and activation time.
Event Monitor Start	event_start	Time when the monitor was started or restarted.
Connection Header	event_connheader	One for each current connection, includes connection time and application name. Event connection headers are only generated for connection, statement, transaction, and deadlock event monitors. Deadlocks with details event monitors produce connection headers only when a deadlock occurs.

## Contents

Information specific to the event monitor's specified event types is presented in the Contents section.

*Table 74. Event Monitor Data Stream: Contents Section*

Event type	Logical data group	Available information
Statement Event	event_stmt	Statement level data, including text for dynamic statements. Statement event monitors do not log fetches.
Subsection Event	event_subsection	Subsection level data.
Transaction Event <sup>1</sup>	event_xact	Transaction level data.
Connection Event	event_conn	Connection level data.
Deadlock Event	event_deadlock	Deadlock level data.
Deadlocked Connection Event	event_dlconn	One for each connection involved in the deadlock, includes applications involved and locks in contention.
Deadlocked Connection Event with Details	event_detailed_dlconn, lock	One for each connection involved in the deadlock, includes applications involved, locks in contention, current statement information, and other locks held by the application contention.
Overflow	event_overflow	Number of records lost - generated when writer cannot keep up with a (non-blocked) event monitor.

Table 74. Event Monitor Data Stream: Contents Section (continued)

Event type	Logical data group	Available information
Deadlocks with details history <sup>2</sup>	event_stmt_history	List of statements executed in any unit of work that was involved in a deadlock.
Deadlocks with details history values <sup>2</sup>	event_data_value	Parameter markers for a statement in the event_stmt_history list.
Activities	event_activity	List of activities that completed executing on the system or were captured before completion.
	event_activystmt	Information about the statement the activity was executing if the activity type was a statement.
	event_activityvals	The data values used as input variables for each activity that is an SQL statement. These data values do not include LOB data, long data, or structured type data.
Statistics	event_scstats	Statistics computed from the activities that executed within each service class, work class, or workload in the system, as well as statistics computed from the threshold queues.
	event_wcstats	
	event_wlstats	
	event_qstats	
	event_histogrambin	
Threshold violations	event_threshold_violations	Information identifying the threshold violated and the time of violation.

<sup>1</sup> This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR UNIT OF WORK statement to monitor transaction events.

<sup>2</sup> This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

## Epilog

The Epilog information is generated during database deactivation (last application finished disconnecting):

Table 75. Event Monitor Data Stream: Epilog Section

Event type	Logical data group	Available information
Database Event	event_db	Database manager level data.
Buffer Pool Event	event_bufferpool	Buffer pool level data.
Table Space Event	event_tablespace	Table space level data.
Table Event	event_table	Table level data.

---

## Event monitor logical data groups and monitor elements

The following table lists the logical data groupings and monitor elements that can be returned by event monitoring.

- “event\_activity logical data group”
- “event\_activitystmt logical data group” on page 281
- “event\_activityvals logical data group” on page 282
- “event\_bufferpool logical data group” on page 282
- “event\_conn logical data group” on page 283
- “event\_connheader logical data group” on page 285
- “event\_connmemuse logical data group” on page 285
- “event\_data\_value logical data group” on page 286
- “event\_db logical data group” on page 286
- “event\_dbheader logical data group” on page 289
- “event\_dbmemuse logical data group” on page 289
- “event\_deadlock logical data group” on page 289
- “event\_detailed\_dlconn logical data group” on page 290
- “event\_dlconn logical data group” on page 291
- “event\_histogrambin logical data group” on page 291
- “event\_log\_header logical data group” on page 292
- “event\_overflow logical data group” on page 292
- “event\_qstats logical data group” on page 292
- “event\_scstats logical data group” on page 292
- “event\_start logical data group” on page 293
- “event\_stmt logical data group” on page 293
- “event\_stmt\_history logical data group” on page 294
- “event\_subsection logical data group” on page 295
- “event\_table logical data group” on page 295
- “event\_tablespace logical data group” on page 296
- “event\_thresholdviolations logical data group” on page 297
- “event\_wlstats logical data group” on page 297
- “event\_wcstats logical data group” on page 298
- “event\_xact logical data group” on page 299
- “lock logical data group” on page 299
- “sqlca logical data group” on page 300

### **event\_activity logical data group**

- “act\_exec\_time - Activity execution time monitor element” on page 305
- “activate\_timestamp - Activate timestamp monitor element” on page 308
- “activity\_id - Activity ID monitor element” on page 309
- “activity\_secondary\_id - Activity secondary ID monitor element” on page 310
- “activity\_type - Activity type monitor element” on page 311
- “address - IP address from which the connection was initiated” on page 312
- “agent\_id - Application handle (agent ID) monitor element” on page 313
- “appl\_id - Application ID” on page 325

"appl\_name - Application name monitor element" on page 328  
 "arm\_correlator - Application response measurement correlator monitor element" on page 335  
 "coord\_partition\_num - Coordinator partition number monitor element" on page 383  
 "db\_work\_action\_set\_id - Database work action set ID monitor element" on page 396  
 "db\_work\_class\_id - Database work class ID monitor element" on page 396  
 details\_xml (This XML document contains all the monitor elements reported in the DETAILS column of the MON\_GET\_ACTIVITY\_DETAILS table function output.)  
 "parent\_activity\_id - Parent activity ID monitor element" on page 538  
 "parent\_uow\_id - Parent unit of work ID monitor element" on page 539  
 "partial\_record - Partial Record monitor element" on page 539  
 "pool\_data\_l\_reads - Buffer pool data logical reads monitor element" on page 556  
 "pool\_data\_p\_reads - Buffer pool data physical reads monitor element" on page 557  
 "num\_remaps - Number of remaps monitor element" on page 525  
 "sc\_work\_action\_set\_id - Service class work action set ID monitor element" on page 630  
 "sc\_work\_class\_id - Service class work class ID monitor element" on page 631  
 "service\_subclass\_name - Service subclass name monitor element" on page 640  
 "service\_superclass\_name - Service superclass name monitor element" on page 640  
 "session\_auth\_id - Session authorization ID monitor element" on page 641  
 "sort\_overflows - Sort overflows monitor element" on page 645  
 "sqlca - SQL Communications Area (SQLCA)" on page 650  
 "time\_completed - Time completed monitor element" on page 707  
 "time\_created - Time created monitor element" on page 707  
 "time\_started - Time started monitor element" on page 708  
 "total\_sort\_time - Total sort time monitor element" on page 720  
 "total\_sorts - Total sorts monitor element" on page 721  
 "tpmon\_acc\_str - TP monitor client accounting string monitor element" on page 726  
 "tpmon\_client\_app - TP monitor client application name monitor element" on page 726  
 "tpmon\_client\_userid - TP monitor client user ID monitor element" on page 727  
 "tpmon\_client\_wkstn - TP monitor client workstation name monitor element" on page 727  
 "uow\_id - Unit of work ID monitor element" on page 734  
 "workload\_id - Workload ID monitor element" on page 745  
 "workload\_occurrence\_id - Workload occurrence identifier monitor element" on page 747  
 "pool\_index\_l\_reads - Buffer pool index logical reads monitor element" on page 564  
 "pool\_index\_p\_reads - Buffer pool index physical reads monitor element" on page 566

“pool\_temp\_data\_l\_reads - Buffer pool temporary data logical reads monitor element” on page 573

“pool\_temp\_data\_p\_reads - Buffer pool temporary data physical reads monitor element” on page 575

“pool\_temp\_index\_l\_reads - Buffer pool temporary index logical reads monitor element” on page 577

“pool\_temp\_index\_p\_reads - Buffer pool temporary index physical reads monitor element” on page 578

“pool\_temp\_xda\_l\_reads - Buffer pool temporary XDA data logical reads monitor element” on page 580

“pool\_temp\_xda\_p\_reads - Buffer pool temporary XDA data physical reads monitor element” on page 582

“pool\_xda\_l\_reads - Buffer pool XDA data logical reads monitor element” on page 585

“pool\_xda\_p\_reads - Buffer pool XDA data physical reads monitor element” on page 588

“prep\_time - Preparation time monitor element” on page 595

“query\_card\_estimate - Query Number of Rows Estimate” on page 603

“query\_cost\_estimate - Query cost estimate monitor element” on page 604

“rows\_fetched - Rows fetched monitor element” on page 622

“rows\_modified - Rows modified monitor element” on page 623

“rows\_returned - Rows returned monitor element” on page 626

“system\_cpu\_time - System CPU Time” on page 677

“user\_cpu\_time - User CPU Time” on page 739

### **event\_activitystmt logical data group**

“activate\_timestamp - Activate timestamp monitor element” on page 308

“activity\_id - Activity ID monitor element” on page 309

“activity\_secondary\_id - Activity secondary ID monitor element” on page 310

“appl\_id - Application ID” on page 325

“comp\_env\_desc - Compilation environment monitor element” on page 362

“creator - Application Creator” on page 387

“eff\_stmt\_text - Effective statement text monitor element” on page 415

“executable\_id - Executable ID monitor element” on page 420

“package\_name - Package name monitor element” on page 534

“package\_version\_id - Package version monitor element” on page 535

“section\_env - Section environment monitor element” on page 632

“section\_number - Section number monitor element” on page 633

“stmt\_first\_use\_time - Statement first use time” on page 661

“stmt\_isolation - Statement isolation” on page 662

“stmt\_last\_use\_time - Statement last use time monitor element” on page 663

“stmt\_lock\_timeout - Statement lock timeout monitor element” on page 663

“stmt\_nest\_level - Statement nesting level monitor element” on page 664

“stmt\_pkgcache\_id - Statement package cache identifier” on page 666

“stmt\_query\_id - Statement query identifier monitor element” on page 666

“stmt\_source\_id - Statement source identifier” on page 667

“stmt\_text - SQL statement text monitor element” on page 669



*"stmt\_type - Statement type monitor element" on page 670*

*"uow\_id - Unit of work ID monitor element" on page 734*

### **event\_activityvals logical data group**

*"activate\_timestamp - Activate timestamp monitor element" on page 308*

*"activity\_id - Activity ID monitor element" on page 309*

*"activity\_secondary\_id - Activity secondary ID monitor element" on page 310*

*"appl\_id - Application ID" on page 325*

*"stmt\_value\_data - Value data" on page 672*

*"stmt\_value\_index - Value index" on page 672*

*"stmt\_value\_isnull - Value has null value monitor element" on page 672*

*"stmt\_value\_isreopt - Variable used for statement reoptimization monitor element" on page 673*

*"stmt\_value\_type - Value type monitor element" on page 673*

*"uow\_id - Unit of work ID monitor element" on page 734*

### **event\_bufferpool logical data group**

*"bp\_id - Buffer pool identifier monitor element" on page 347*

*"bp\_name - Buffer pool name monitor element" on page 347*

*"db\_name - Database Name" on page 393*

*"db\_path - Database Path" on page 394*

*"direct\_read\_reqs - Direct read requests monitor element" on page 404*

*"direct\_read\_time - Direct read time monitor element" on page 406*

*"direct\_reads - Direct reads from database monitor element" on page 407*

*"direct\_write\_reqs - Direct write requests monitor element" on page 409*

*"direct\_write\_time - Direct write time monitor element" on page 410*

*"direct\_writes - Direct writes to database monitor element" on page 412*

*"event\_time - Event Time" on page 419*

*"evmon\_activates - Number of Event Monitor Activations" on page 419*

*"evmon\_flushes - Number of Event Monitor Flushes" on page 420*

*"files\_closed - Database files closed monitor element" on page 438*

*"partial\_record - Partial Record monitor element" on page 539*

*"pool\_async\_data\_read\_reqs - Buffer pool asynchronous read requests monitor element" on page 546*

*"pool\_async\_data\_reads - Buffer pool asynchronous data reads monitor element" on page 547*

*"pool\_async\_data\_writes - Buffer pool asynchronous data writes monitor element" on page 547*

*"pool\_async\_index\_reads - Buffer pool asynchronous index reads monitor element" on page 549*

*"pool\_async\_index\_writes - Buffer pool asynchronous index writes monitor element" on page 550*

*"pool\_async\_read\_time - Buffer Pool Asynchronous Read Time" on page 551*

*"pool\_async\_write\_time - Buffer Pool Asynchronous Write Time" on page 551*

*"pool\_data\_l\_reads - Buffer pool data logical reads monitor element" on page 556*

"pool\_data\_p\_reads - Buffer pool data physical reads monitor element" on page 557  
"pool\_data\_writes - Buffer pool data writes monitor element" on page 559  
"pool\_index\_l\_reads - Buffer pool index logical reads monitor element" on page 564  
"pool\_index\_p\_reads - Buffer pool index physical reads monitor element" on page 566  
"pool\_index\_writes - Buffer pool index writes monitor element" on page 567  
"pool\_read\_time - Total buffer pool physical read time monitor element" on page 571  
"pool\_write\_time - Total buffer pool physical write time monitor element" on page 584

### **event\_conn logical data group**

"acc\_curs\_blk - Accepted Block Cursor Requests" on page 302  
"agent\_id - Application handle (agent ID) monitor element" on page 313  
"appl\_id - Application ID" on page 325  
"appl\_priority - Application Agent Priority" on page 329  
"appl\_priority\_type - Application Priority Type" on page 329  
"appl\_section\_inserts - Section Inserts monitor element" on page 330  
"appl\_section\_lookups - Section Lookups" on page 330  
"authority\_bitmap - User authorization level monitor element" on page 341  
"authority\_lvl - User authorization level monitor element" on page 342  
"binds\_precompiles - Binds/Precompiles Attempted" on page 344  
"cat\_cache\_inserts - Catalog Cache Inserts" on page 349  
"cat\_cache\_lookups - Catalog Cache Lookups" on page 350  
"cat\_cache\_overflows - Catalog Cache Overflows" on page 351  
"commit\_sql\_stmts - Commit Statements Attempted" on page 361  
"ddl\_sql\_stmts - Data Definition Language (DDL) SQL Statements" on page 397  
"deadlocks - Deadlocks detected monitor element" on page 399  
"direct\_read\_reqs - Direct read requests monitor element" on page 404  
"direct\_read\_time - Direct read time monitor element" on page 406  
"direct\_reads - Direct reads from database monitor element" on page 407  
"direct\_write\_reqs - Direct write requests monitor element" on page 409  
"direct\_write\_time - Direct write time monitor element" on page 410  
"direct\_writes - Direct writes to database monitor element" on page 412  
"disconn\_time - Database Deactivation Timestamp" on page 413  
"dynamic\_sql\_stmts - Dynamic SQL Statements Attempted" on page 414  
"failed\_sql\_stmts - Failed Statement Operations" on page 421  
"hash\_join\_overflows - Hash Join Overflows" on page 455  
"hash\_join\_small\_overflows - Hash Join Small Overflows" on page 455  
"int\_auto\_rebinds - Internal Automatic Rebinds" on page 463  
"int\_commits - Internal Commits" on page 463  
"int\_deadlock\_rollback - Internal Rollbacks Due To Deadlock" on page 464  
"int\_rollback - Internal Rollbacks" on page 465  
"int\_rows\_deleted - Internal Rows Deleted" on page 466

"int\_rows\_inserted - Internal Rows Inserted" on page 466  
"int\_rows\_updated - Internal Rows Updated" on page 467  
"lock\_escalation - Lock escalation monitor element" on page 480  
"lock\_timeouts - Number of lock timeouts monitor element" on page 488  
"lock\_wait\_time - Time waited on locks monitor element" on page 490  
"lock\_waits - Lock waits monitor element" on page 492  
"olap\_func\_overflows - OLAP Function Overflows monitor element" on page 527  
"partial\_record - Partial Record monitor element" on page 539  
"int\_rows\_updated - Internal Rows Updated" on page 467  
"pkg\_cache\_inserts - Package Cache Inserts" on page 543  
"pkg\_cache\_lookups - Package Cache Lookups" on page 543  
"pool\_data\_l\_reads - Buffer pool data logical reads monitor element" on page 556  
"pool\_data\_p\_reads - Buffer pool data physical reads monitor element" on page 557  
"pool\_data\_writes - Buffer pool data writes monitor element" on page 559  
"pool\_index\_l\_reads - Buffer pool index logical reads monitor element" on page 564  
"pool\_index\_p\_reads - Buffer pool index physical reads monitor element" on page 566  
"pool\_index\_writes - Buffer pool index writes monitor element" on page 567  
"pool\_read\_time - Total buffer pool physical read time monitor element" on page 571  
"pool\_temp\_data\_l\_reads - Buffer pool temporary data logical reads monitor element" on page 573  
"pool\_temp\_data\_p\_reads - Buffer pool temporary data physical reads monitor element" on page 575  
"pool\_temp\_index\_l\_reads - Buffer pool temporary index logical reads monitor element" on page 577  
"pool\_temp\_index\_p\_reads - Buffer pool temporary index physical reads monitor element" on page 578  
"pool\_write\_time - Total buffer pool physical write time monitor element" on page 584  
"prefetch\_wait\_time - Time waited for prefetch monitor element" on page 595  
"priv\_workspace\_num\_overflows - Private Workspace Overflows" on page 597  
"priv\_workspace\_section\_inserts - Private Workspace Section Inserts" on page 597  
"priv\_workspace\_section\_lookups - Private Workspace Section Lookups" on page 598  
"priv\_workspace\_size\_top - Maximum Private Workspace Size" on page 599  
"rej\_curs\_blk - Rejected Block Cursor Requests" on page 609  
"rollback\_sql\_stmts - Rollback Statements Attempted" on page 619  
"rows\_read - Rows read monitor element" on page 624  
"rows\_selected - Rows Selected" on page 628  
"rows\_written - Rows Written" on page 629  
"select\_sql\_stmts - Select SQL Statements Executed" on page 634

"sequence\_no - Sequence number monitor element" on page 635  
 "shr\_workspace\_num\_overflows - Shared Workspace Overflows" on page 642  
 "shr\_workspace\_section\_inserts - Shared Workspace Section Inserts" on page 642  
 "shr\_workspace\_section\_lookups - Shared Workspace Section Lookups" on page 643  
 "shr\_workspace\_size\_top - Maximum Shared Workspace Size" on page 643  
 "sort\_overflows - Sort overflows monitor element" on page 645  
 "static\_sql\_stmts - Static SQL Statements Attempted" on page 657  
 "system\_cpu\_time - System CPU Time" on page 677  
 "total\_hash\_joins - Total Hash Joins" on page 713  
 "total\_hash\_loops - Total Hash Loops" on page 714  
 "total\_olap\_funcs - Total OLAP Functions monitor element" on page 715  
 "total\_sec\_cons - Secondary Connections" on page 717  
 "total\_sort\_time - Total sort time monitor element" on page 720  
 "total\_sorts - Total sorts monitor element" on page 721  
 "uid\_sql\_stmts - Update/Insert/Delete SQL Statements Executed" on page 732  
 "unread\_prefetch\_pages - Unread prefetch pages monitor element" on page 733  
 "user\_cpu\_time - User CPU Time" on page 739  
 "x\_lock\_escals - Exclusive Lock Escalations" on page 748  
 "xquery\_stmts - XQuery Statements Attempted" on page 750

### **event\_connheader logical data group**

"agent\_id - Application handle (agent ID) monitor element" on page 313  
 "appl\_id - Application ID" on page 325  
 "appl\_name - Application name monitor element" on page 328  
 "auth\_id - Authorization ID" on page 340  
 "client\_db\_alias - Database Alias Used by Application" on page 355  
 "client\_pid - Client Process ID" on page 356  
 "client\_platform - Client Operating Platform" on page 357  
 "client\_prdid - Client product and version ID monitor element" on page 357  
 "client\_protocol - Client Communication Protocol" on page 358  
 "codepage\_id - ID of Code Page Used by Application" on page 360  
 "conn\_time - Time of database connection monitor element" on page 371  
 "corr\_token - DRDA Correlation Token" on page 383  
 "execution\_id - User Login ID" on page 421  
 "node\_number - Node Number" on page 517  
 "sequence\_no - Sequence number monitor element" on page 635  
 "territory\_code - Database Territory Code" on page 704

### **event\_connmemuse logical data group**

"node\_number - Node Number" on page 517  
 "pool\_config\_size - Configured Size of Memory Pool" on page 554  
 "pool\_cur\_size - Current Size of Memory Pool" on page 555  
 "pool\_id - Memory Pool Identifier" on page 563  
 "pool\_secondary\_id - Memory Pool Secondary Identifier" on page 573

*"pool\_watermark - Memory Pool Watermark" on page 583*

### **event\_data\_value logical data group**

*"deadlock\_id - Deadlock Event Identifier" on page 398*

*"deadlock\_node - Partition Number Where Deadlock Occurred" on page 399*

*"evmon\_activates - Number of Event Monitor Activations" on page 419*

*"participant\_no - Participant within Deadlock" on page 540*

*"stmt\_history\_id - Statement history identifier" on page 661*

*"stmt\_value\_data - Value data" on page 672*

*"stmt\_value\_index - Value index" on page 672*

*"stmt\_value\_isnull - Value has null value monitor element" on page 672*

*"stmt\_value\_isreopt - Variable used for statement reoptimization monitor element" on page 673*

*"stmt\_value\_type - Value type monitor element" on page 673*

### **event\_db logical data group**

*"active\_hash\_joins - Active hash joins" on page 308*

*"appl\_section\_inserts - Section Inserts monitor element" on page 330*

*"appl\_section\_lookups - Section Lookups" on page 330*

*"async\_runstats - Total number of asynchronous RUNSTATS requests monitor element" on page 335*

*"binds\_precompiles - Binds/Precompiles Attempted" on page 344*

*"blocks\_pending\_cleanup - Pending cleanup rolled-out blocks monitor element" on page 346*

*"cat\_cache\_inserts - Catalog Cache Inserts" on page 349*

*"cat\_cache\_lookups - Catalog Cache Lookups" on page 350*

*"cat\_cache\_overflows - Catalog Cache Overflows" on page 351*

*"cat\_cache\_size\_top - Catalog cache high watermark monitor element" on page 351*

*"catalog\_node - Catalog Node Number" on page 352*

*"catalog\_node\_name - Catalog Node Network Name" on page 352*

*"commit\_sql\_stmts - Commit Statements Attempted" on page 361*

*"connections\_top - Maximum Number of Concurrent Connections" on page 372*

*"db\_heap\_top - Maximum Database Heap Allocated" on page 392*

*"ddl\_sql\_stmts - Data Definition Language (DDL) SQL Statements" on page 397*

*"deadlocks - Deadlocks detected monitor element" on page 399*

*"direct\_read\_reqs - Direct read requests monitor element" on page 404*

*"direct\_read\_time - Direct read time monitor element" on page 406*

*"direct\_reads - Direct reads from database monitor element" on page 407*

*"direct\_write\_reqs - Direct write requests monitor element" on page 409*

*"direct\_write\_time - Direct write time monitor element" on page 410*

*"direct\_writes - Direct writes to database monitor element" on page 412*

*"disconn\_time - Database Deactivation Timestamp" on page 413*

*"dynamic\_sql\_stmts - Dynamic SQL Statements Attempted" on page 414*

*"evmon\_activates - Number of Event Monitor Activations" on page 419*

*"evmon\_flushes - Number of Event Monitor Flushes" on page 420*

“failed\_sql\_stmts - Failed Statement Operations” on page 421  
“files\_closed - Database files closed monitor element” on page 438  
“hash\_join\_overflows - Hash Join Overflows” on page 455  
“hash\_join\_small\_overflows - Hash Join Small Overflows” on page 455  
“int\_auto\_rebinds - Internal Automatic Rebinds” on page 463  
“int\_commits - Internal Commits” on page 463  
“int\_rollbacks - Internal Rollbacks” on page 465  
“int\_rows\_deleted - Internal Rows Deleted” on page 466  
“int\_rows\_inserted - Internal Rows Inserted” on page 466  
“int\_rows\_updated - Internal Rows Updated” on page 467  
“lock\_escals - Number of lock escalations monitor element” on page 480  
“lock\_timeouts - Number of lock timeouts monitor element” on page 488  
“lock\_wait\_time - Time waited on locks monitor element” on page 490  
“lock\_waits - Lock waits monitor element” on page 492  
“log\_held\_by\_dirty\_pages - Amount of Log Space Accounted for by Dirty Pages” on page 498  
“log\_read\_time - Log Read Time” on page 499  
“log\_reads - Number of Log Pages Read” on page 499  
“log\_to\_redo\_for\_recovery - Amount of Log to be Redone for Recovery” on page 500  
“log\_write\_time - Log Write Time” on page 500  
“log\_writes - Number of Log Pages Written” on page 500  
“num\_log\_read\_io - Number of Log Reads” on page 524  
“num\_log\_write\_io - Number of Log Writes” on page 524  
“num\_threshold\_violations - Number of threshold violations monitor element” on page 525  
“olap\_func\_overflows - OLAP Function Overflows monitor element” on page 527  
“partial\_record - Partial Record monitor element” on page 539  
“pkg\_cache\_inserts - Package Cache Inserts” on page 543  
“pkg\_cache\_lookups - Package Cache Lookups” on page 543  
“pkg\_cache\_num\_overflows - Package Cache Overflows” on page 545  
“pkg\_cache\_size\_top - Package cache high watermark” on page 545  
“pool\_async\_data\_read\_reqs - Buffer pool asynchronous read requests monitor element” on page 546  
“pool\_async\_data\_reads - Buffer pool asynchronous data reads monitor element” on page 547  
“pool\_async\_data\_writes - Buffer pool asynchronous data writes monitor element” on page 547  
“pool\_async\_index\_read\_reqs - Buffer pool asynchronous index read requests monitor element” on page 548  
“pool\_async\_index\_reads - Buffer pool asynchronous index reads monitor element” on page 549  
“pool\_async\_index\_writes - Buffer pool asynchronous index writes monitor element” on page 550  
“pool\_async\_read\_time - Buffer Pool Asynchronous Read Time” on page 551  
“pool\_async\_write\_time - Buffer Pool Asynchronous Write Time” on page 551

"pool\_data\_l\_reads - Buffer pool data logical reads monitor element" on page 556

"pool\_data\_p\_reads - Buffer pool data physical reads monitor element" on page 557

"pool\_data\_writes - Buffer pool data writes monitor element" on page 559

"pool\_drty\_pg\_steal\_clns - Buffer pool victim page cleaners triggered monitor element" on page 561

"pool\_drty\_pg\_thrsh\_clns - Buffer pool threshold cleaners triggered monitor element" on page 562

"pool\_index\_l\_reads - Buffer pool index logical reads monitor element" on page 564

"pool\_index\_p\_reads - Buffer pool index physical reads monitor element" on page 566

"pool\_index\_writes - Buffer pool index writes monitor element" on page 567

"pool\_lsn\_gap\_clns - Buffer pool log space cleaners triggered monitor element" on page 569

"pool\_no\_victim\_buffer - Buffer pool no victim buffers monitor element" on page 570

"pool\_read\_time - Total buffer pool physical read time monitor element" on page 571

"pool\_temp\_data\_l\_reads - Buffer pool temporary data logical reads monitor element" on page 573

"pool\_temp\_data\_p\_reads - Buffer pool temporary data physical reads monitor element" on page 575

"pool\_temp\_index\_l\_reads - Buffer pool temporary index logical reads monitor element" on page 577

"pool\_temp\_index\_p\_reads - Buffer pool temporary index physical reads monitor element" on page 578

"pool\_write\_time - Total buffer pool physical write time monitor element" on page 584

"post\_shrthreshold\_hash\_joins - Post threshold hash joins" on page 591

"post\_shrthreshold\_sorts - Post shared threshold sorts monitor element" on page 591

"prefetch\_wait\_time - Time waited for prefetch monitor element" on page 595

"priv\_workspace\_num\_overflows - Private Workspace Overflows" on page 597

"priv\_workspace\_section\_inserts - Private Workspace Section Inserts" on page 597

"priv\_workspace\_section\_lookups - Private Workspace Section Lookups" on page 598

"priv\_workspace\_size\_top - Maximum Private Workspace Size" on page 599

"rollback\_sql\_stmts - Rollback Statements Attempted" on page 619

"rows\_deleted - Rows deleted monitor element" on page 621

"rows\_inserted - Rows inserted monitor element" on page 622

"rows\_read - Rows read monitor element" on page 624

"rows\_selected - Rows Selected" on page 628

"rows\_updated - Rows updated monitor element" on page 628

"sec\_log\_used\_top - Maximum Secondary Log Space Used" on page 631

"select\_sql\_stmts - Select SQL Statements Executed" on page 634



“server\_platform - Server Operating System” on page 637  
 “shr\_workspace\_num\_overflows - Shared Workspace Overflows” on page 642  
 “shr\_workspace\_section\_inserts - Shared Workspace Section Inserts” on page 642  
 “shr\_workspace\_section\_lookups - Shared Workspace Section Lookups” on page 643  
 “shr\_workspace\_size\_top - Maximum Shared Workspace Size” on page 643  
 “sort\_overflows - Sort overflows monitor element” on page 645  
 “static\_sql\_stmts - Static SQL Statements Attempted” on page 657  
 “stats\_cache\_size - Size of statistics cache monitor element” on page 658  
 “stats\_fabricate\_time - Total time spent on statistics fabrication activities monitor element” on page 659  
 “stats\_fabrications - Total number of statistics fabrications monitor elements” on page 660  
 “sync\_runstats - Total number of synchronous RUNSTATS activities monitor element” on page 675  
 “sync\_runstats\_time - Total time spent on synchronous RUNSTATS activities monitor element” on page 676  
 “tot\_log\_used\_top - Maximum Total Log Space Used” on page 709  
 “total\_cons - Connects Since Database Activation” on page 711  
 “total\_hash\_joins - Total Hash Joins” on page 713  
 “total\_hash\_loops - Total Hash Loops” on page 714  
 “total\_olap\_funcs - Total OLAP Functions monitor element” on page 715  
 “total\_sort\_time - Total sort time monitor element” on page 720  
 “total\_sorts - Total sorts monitor element” on page 721  
 “uid\_sql\_stmts - Update/Insert/Delete SQL Statements Executed” on page 732  
 “unread\_prefetch\_pages - Unread prefetch pages monitor element” on page 733  
 “x\_lock\_escals - Exclusive Lock Escalations” on page 748  
 “xquery\_stmts - XQuery Statements Attempted” on page 750

### **event\_dbheader logical data group**

“conn\_time - Time of database connection monitor element” on page 371  
 “db\_name - Database Name” on page 393  
 “db\_path - Database Path” on page 394

### **event\_dbmemuse logical data group**

“node\_number - Node Number” on page 517  
 “pool\_config\_size - Configured Size of Memory Pool” on page 554  
 “pool\_cur\_size - Current Size of Memory Pool” on page 555  
 “pool\_id - Memory Pool Identifier” on page 563  
 “pool\_watermark - Memory Pool Watermark” on page 583

### **event\_deadlock logical data group**

“deadlock\_id - Deadlock Event Identifier” on page 398  
 “deadlock\_node - Partition Number Where Deadlock Occurred” on page 399  
 “dl\_conns - Connections involved in deadlock monitor element” on page 414  
 “evmon\_activates - Number of Event Monitor Activations” on page 419



"rolled\_back\_agent\_id - Rolled Back Agent" on page 620  
"rolled\_back\_appl\_id - Rolled Back Application" on page 620  
"rolled\_back\_participant\_no - Rolled back application participant monitor element" on page 620  
"rolled\_back\_sequence\_no - Rolled Back Sequence Number" on page 621  
"start\_time - Event Start Time" on page 657

### **event\_detailed\_dlconn logical data group**

"agent\_id - Application handle (agent ID) monitor element" on page 313  
"appl\_id - Application ID" on page 325  
"appl\_id\_holding\_lk - Application ID Holding Lock" on page 327  
"blocking\_cursor - Blocking Cursor" on page 345  
"consistency\_token - Package consistency token monitor element" on page 373  
"creator - Application Creator" on page 387  
"cursor\_name - Cursor Name" on page 389  
"data\_partition\_id - Data partition identifier monitor element" on page 390  
"deadlock\_id - Deadlock Event Identifier" on page 398  
"deadlock\_node - Partition Number Where Deadlock Occurred" on page 399  
"evmon\_activates - Number of Event Monitor Activations" on page 419  
"lock\_escalation - Lock escalation monitor element" on page 480  
"lock\_mode - Lock mode monitor element" on page 483  
"lock\_mode\_requested - Lock mode requested monitor element" on page 484  
"lock\_node - Lock Node" on page 485  
"lock\_object\_name - Lock Object Name" on page 485  
"lock\_object\_type - Lock object type waited on monitor element" on page 486  
"lock\_wait\_start\_time - Lock Wait Start Timestamp" on page 490  
"locks\_held - Locks Held" on page 494  
"locks\_in\_list - Number of Locks Reported" on page 495  
"package\_name - Package name monitor element" on page 534  
"package\_version\_id - Package version monitor element" on page 535  
"participant\_no - Participant within Deadlock" on page 540  
"participant\_no\_holding\_lk - Participant Holding a Lock on the Object Required by Application" on page 540  
"section\_number - Section number monitor element" on page 633  
"sequence\_no - Sequence number monitor element" on page 635  
"sequence\_no\_holding\_lk - Sequence Number Holding Lock" on page 636  
"start\_time - Event Start Time" on page 657  
"stmt\_operation/operation - Statement operation monitor element" on page 664  
"stmt\_text - SQL statement text monitor element" on page 669  
"stmt\_type - Statement type monitor element" on page 670  
"table\_name - Table name monitor element" on page 678  
"table\_schema - Table schema name monitor element" on page 679  
"tablespace\_name - Table space name monitor element" on page 687

## **event\_dlconn logical data group**

- “agent\_id - Application handle (agent ID) monitor element” on page 313
- “appl\_id - Application ID” on page 325
- “appl\_id\_holding\_lk - Application ID Holding Lock” on page 327
- “data\_partition\_id - Data partition identifier monitor element” on page 390
- “deadlock\_id - Deadlock Event Identifier” on page 398
- “deadlock\_node - Partition Number Where Deadlock Occurred” on page 399
- “evmon\_activates - Number of Event Monitor Activations” on page 419
- “lock\_attributes - Lock attributes monitor element” on page 477
- “lock\_count - Lock count monitor element” on page 478
- “lock\_current\_mode - Original Lock Mode Before Conversion” on page 479
- “lock\_escalation - Lock escalation monitor element” on page 480
- “lock\_hold\_count - Lock hold count monitor element” on page 482
- “lock\_mode - Lock mode monitor element” on page 483
- “lock\_mode\_requested - Lock mode requested monitor element” on page 484
- “lock\_name - Lock name monitor element” on page 484
- “lock\_node - Lock Node” on page 485
- “lock\_object\_name - Lock Object Name” on page 485
- “lock\_object\_type - Lock object type waited on monitor element” on page 486
- “lock\_release\_flags - Lock release flags monitor element” on page 486
- “lock\_wait\_start\_time - Lock Wait Start Timestamp” on page 490
- “participant\_no - Participant within Deadlock” on page 540
- “participant\_no\_holding\_lk - Participant Holding a Lock on the Object Required by Application” on page 540
- “sequence\_no - Sequence number monitor element” on page 635
- “sequence\_no\_holding\_lk - Sequence Number Holding Lock” on page 636
- “start\_time - Event Start Time” on page 657
- “table\_name - Table name monitor element” on page 678
- “table\_schema - Table schema name monitor element” on page 679
- “tablespace\_name - Table space name monitor element” on page 687
- “tpmon\_acc\_str - TP monitor client accounting string monitor element” on page 726
- “tpmon\_client\_app - TP monitor client application name monitor element” on page 726
- “tpmon\_client\_userid - TP monitor client user ID monitor element” on page 727
- “tpmon\_client\_wkstn - TP monitor client workstation name monitor element” on page 727

## **event\_histogrambin logical data group**

- “bin\_id - Histogram bin identifier monitor element” on page 343
- “bottom - Histogram bin bottom monitor element” on page 346
- “histogram\_type - Histogram type monitor element” on page 456
- “number\_in\_bin - Number in bin monitor element” on page 526
- “service\_class\_id - Service class ID monitor element” on page 639
- “statistics\_timestamp - Statistics timestamp monitor element” on page 658
- “top - Histogram bin top monitor element” on page 708

*"work\_action\_set\_id - Work action set ID monitor element" on page 744*

*"work\_class\_id - Work class ID monitor element" on page 745*

*"workload\_id - Workload ID monitor element" on page 745*

### **event\_log\_header logical data group**

*"byte\_order - Byte Order of Event Data" on page 349*

*"codepage\_id - ID of Code Page Used by Application" on page 360*

*"event\_monitor\_name - Event Monitor Name" on page 419*

*"num\_nodes\_in\_db2\_instance - Number of Nodes in Partition" on page 524*

*"server\_instance\_name - Server Instance Name" on page 637*

*"server\_prdid - Server Product/Version ID" on page 637*

*"territory\_code - Database Territory Code" on page 704*

*"version - Version of Monitor Data" on page 742*

### **event\_overflow logical data group**

*"count - Number of Event Monitor Overflows" on page 384*

*"first\_overflow\_time - Time of First Event Overflow" on page 440*

*"last\_overflow\_time - Time of Last Event Overflow" on page 474*

*"node\_number - Node Number" on page 517*

### **event\_qstats logical data group**

*"last\_wlm\_reset - Time of last reset monitor element" on page 475*

*"queue\_assignments\_total - Queue assignments total monitor element" on page 604*

*"queue\_size\_top - Queue size top monitor element" on page 605*

*"queue\_time\_total - Queue time total monitor element" on page 605*

*"service\_subclass\_name - Service subclass name monitor element" on page 640*

*"service\_superclass\_name - Service superclass name monitor element" on page 640*

*"statistics\_timestamp - Statistics timestamp monitor element" on page 658*

*"threshold\_domain - Threshold domain monitor element" on page 705*

*"threshold\_name - Threshold name monitor element" on page 705*

*"threshold\_predicate - Threshold predicate monitor element" on page 706*

*"thresholdid - Threshold ID monitor element" on page 706*

*"work\_action\_set\_name - Work action set name monitor element" on page 745*

*"work\_class\_name - Work class name monitor element" on page 745*

### **event\_scstats logical data group**

*"act\_cpu\_time\_top - Activity CPU time top monitor element" on page 304*

*"act\_remapped\_in - Activities remapped in monitor element" on page 306*

*"act\_remapped\_out - Activities remapped out monitor element" on page 306*

*"act\_rows\_read\_top - Activity rows read top monitor element" on page 307*

*"agg\_temp\_tablespace\_top - Aggregate temporary table space top monitor element" on page 322*

*"concurrent\_act\_top - Concurrent activity top monitor element" on page 363*

*"concurrent\_wlo\_top - Concurrent workload occurrences top monitor element" on page 364*

"concurrent\_connection\_top - Concurrent connection top monitor element" on page 364  
 "coord\_act\_aborted\_total - Coordinator activities aborted total monitor element" on page 376  
 "coord\_act\_completed\_total - Coordinator activities completed total monitor element" on page 376  
 "coord\_act\_est\_cost\_avg - Coordinator activity estimated cost average monitor element" on page 377  
 "coord\_act\_exec\_time\_avg - Coordinator activities execution time average monitor element" on page 378  
 "coord\_act\_interarrival\_time\_avg - Coordinator activity arrival time average monitor element" on page 378  
 "coord\_act\_lifetime\_avg - Coordinator activity lifetime average monitor element" on page 379  
 "coord\_act\_lifetime\_top - Coordinator activity lifetime top monitor element" on page 380  
 "coord\_act\_queue\_time\_avg - Coordinator activity queue time average monitor element" on page 381  
 "coord\_act\_rejected\_total - Coordinator activities rejected total monitor element" on page 382  
 "cost\_estimate\_top - Cost estimate top monitor element" on page 384  
 details\_xml (This XML document contains all the monitor elements reported in the DETAILS column of the MON\_GET\_SERVICE\_SUBCLASS\_DETAILS table function output.)  
 "last\_wlm\_reset - Time of last reset monitor element" on page 475  
 "request\_exec\_time\_avg - Request execution time average monitor element" on page 617  
 "rows\_returned\_top - Actual rows returned top monitor element" on page 627  
 "service\_class\_id - Service class ID monitor element" on page 639  
 "service\_subclass\_name - Service subclass name monitor element" on page 640  
 "service\_superclass\_name - Service superclass name monitor element" on page 640  
 "statistics\_timestamp - Statistics timestamp monitor element" on page 658  
 "temp\_tablespace\_top - Temporary table space top monitor element" on page 703  
 "uow\_total\_time\_top - UOW total time top monitor element" on page 738

### **event\_start logical data group**

"start\_time - Event Start Time" on page 657

### **event\_stmt logical data group**

"agent\_id - Application handle (agent ID) monitor element" on page 313  
 "agents\_top - Number of Agents Created" on page 321  
 "appl\_id - Application ID" on page 325  
 "blocking\_cursor - Blocking Cursor" on page 345  
 "consistency\_token - Package consistency token monitor element" on page 373  
 "creator - Application Creator" on page 387  
 "cursor\_name - Cursor Name" on page 389  
 "fetch\_count - Number of Successful Fetches" on page 438

"int\_rows\_deleted - Internal Rows Deleted" on page 466  
"int\_rows\_inserted - Internal Rows Inserted" on page 466  
"int\_rows\_updated - Internal Rows Updated" on page 467  
"package\_name - Package name monitor element" on page 534  
"package\_version\_id - Package version monitor element" on page 535  
"partial\_record - Partial Record monitor element" on page 539  
"pool\_data\_l\_reads - Buffer pool data logical reads monitor element" on page 556  
"pool\_data\_p\_reads - Buffer pool data physical reads monitor element" on page 557  
"pool\_index\_l\_reads - Buffer pool index logical reads monitor element" on page 564  
"pool\_index\_p\_reads - Buffer pool index physical reads monitor element" on page 566  
"pool\_temp\_data\_l\_reads - Buffer pool temporary data logical reads monitor element" on page 573  
"pool\_temp\_data\_p\_reads - Buffer pool temporary data physical reads monitor element" on page 575  
"pool\_temp\_index\_l\_reads - Buffer pool temporary index logical reads monitor element" on page 577  
"pool\_temp\_index\_p\_reads - Buffer pool temporary index physical reads monitor element" on page 578  
"rows\_read - Rows read monitor element" on page 624  
"rows\_written - Rows Written" on page 629  
"section\_number - Section number monitor element" on page 633  
"sequence\_no - Sequence number monitor element" on page 635  
"sort\_overflows - Sort overflows monitor element" on page 645  
"sql\_req\_id - Request Identifier for SQL Statement" on page 649  
"sqlca - SQL Communications Area (SQLCA)" on page 650  
"start\_time - Event Start Time" on page 657  
"stats\_fabricate\_time - Total time spent on statistics fabrication activities monitor element" on page 659  
"stmt\_operation/operation - Statement operation monitor element" on page 664  
"stmt\_text - SQL statement text monitor element" on page 669  
"stmt\_type - Statement type monitor element" on page 670  
"stop\_time - Event Stop Time" on page 674  
"sync\_runstats\_time - Total time spent on synchronous RUNSTATS activities monitor element" on page 676  
"system\_cpu\_time - System CPU Time" on page 677  
"total\_sort\_time - Total sort time monitor element" on page 720  
"total\_sorts - Total sorts monitor element" on page 721  
"user\_cpu\_time - User CPU Time" on page 739

### **event\_stmt\_history logical data group**

"comp\_env\_desc - Compilation environment monitor element" on page 362  
"creator - Application Creator" on page 387  
"deadlock\_id - Deadlock Event Identifier" on page 398

*"deadlock\_node - Partition Number Where Deadlock Occurred"* on page 399  
*"evmon\_activates - Number of Event Monitor Activations"* on page 419  
*"package\_name - Package name monitor element"* on page 534  
*"package\_version\_id - Package version monitor element"* on page 535  
*"participant\_no - Participant within Deadlock"* on page 540  
*"section\_number - Section number monitor element"* on page 633  
*"sequence\_no - Sequence number monitor element"* on page 635  
*"stmt\_first\_use\_time - Statement first use time"* on page 661  
*"stmt\_history\_id - Statement history identifier"* on page 661  
*"stmt\_invocation\_id - Statement invocation identifier monitor element"* on page 662  
*"stmt\_isolation - Statement isolation"* on page 662  
*"stmt\_last\_use\_time - Statement last use time monitor element"* on page 663  
*"stmt\_lock\_timeout - Statement lock timeout monitor element"* on page 663  
*"stmt\_nest\_level - Statement nesting level monitor element"* on page 664  
*"stmt\_pkgcache\_id - Statement package cache identifier"* on page 666  
*"stmt\_query\_id - Statement query identifier monitor element"* on page 666  
*"stmt\_source\_id - Statement source identifier"* on page 667  
*"stmt\_text - SQL statement text monitor element"* on page 669  
*"stmt\_type - Statement type monitor element"* on page 670

### **event\_subsection logical data group**

*"agent\_id - Application handle (agent ID) monitor element"* on page 313  
*"num\_agents - Number of Agents Working on a Statement"* on page 518  
*"partial\_record - Partial Record monitor element"* on page 539  
*"ss\_exec\_time - Subsection Execution Elapsed Time"* on page 655  
*"ss\_node\_number - Subsection Node Number"* on page 655  
*"ss\_number - Subsection Number"* on page 655  
*"ss\_sys\_cpu\_time - System CPU Time used by Subsection"* on page 656  
*"ss\_usr\_cpu\_time - User CPU Time used by Subsection"* on page 656  
*"tq\_max\_send\_spills - Maximum number of table queue buffers overflows"* on page 729  
*"tq\_rows\_read - Number of Rows Read from table queues"* on page 729  
*"tq\_rows\_written - Number of rows written to table queues"* on page 730  
*"tq\_tot\_send\_spills - Total number of table queue buffers overflowed monitor element"* on page 730

### **event\_table logical data group**

*"data\_object\_pages - Data Object Pages"* on page 390  
*"data\_partition\_id - Data partition identifier monitor element"* on page 390  
*"event\_time - Event Time"* on page 419  
*"evmon\_activates - Number of Event Monitor Activations"* on page 419  
*"evmon\_flushes - Number of Event Monitor Flushes"* on page 420  
*"index\_object\_pages - Index Object Pages"* on page 460  
*"lob\_object\_pages - LOB Object Pages"* on page 476  
*"long\_object\_pages - Long Object Pages"* on page 501

"overflow\_accesses - Accesses to overflowed records monitor element" on page 533  
"page\_reorgs - Page Reorganizations" on page 536  
"partial\_record - Partial Record monitor element" on page 539  
"rows\_read - Rows read monitor element" on page 624  
"rows\_written - Rows Written" on page 629  
"table\_name - Table name monitor element" on page 678  
"table\_schema - Table schema name monitor element" on page 679  
"table\_type - Table type monitor element" on page 681

### **event\_tablespace logical data group**

"direct\_read\_reqs - Direct read requests monitor element" on page 404  
"direct\_read\_time - Direct read time monitor element" on page 406  
"direct\_reads - Direct reads from database monitor element" on page 407  
"direct\_write\_reqs - Direct write requests monitor element" on page 409  
"direct\_write\_time - Direct write time monitor element" on page 410  
"direct\_writes - Direct writes to database monitor element" on page 412  
"event\_time - Event Time" on page 419  
"evmon\_activates - Number of Event Monitor Activations" on page 419  
"evmon\_flushes - Number of Event Monitor Flushes" on page 420  
"files\_closed - Database files closed monitor element" on page 438  
"partial\_record - Partial Record monitor element" on page 539  
"pool\_async\_data\_read\_reqs - Buffer pool asynchronous read requests monitor element" on page 546  
"pool\_async\_data\_reads - Buffer pool asynchronous data reads monitor element" on page 547  
"pool\_async\_data\_writes - Buffer pool asynchronous data writes monitor element" on page 547  
"pool\_async\_index\_read\_reqs - Buffer pool asynchronous index read requests monitor element" on page 548  
"pool\_async\_index\_reads - Buffer pool asynchronous index reads monitor element" on page 549  
"pool\_async\_index\_writes - Buffer pool asynchronous index writes monitor element" on page 550  
"pool\_async\_read\_time - Buffer Pool Asynchronous Read Time" on page 551  
"pool\_async\_write\_time - Buffer Pool Asynchronous Write Time" on page 551  
"pool\_data\_l\_reads - Buffer pool data logical reads monitor element" on page 556  
"pool\_data\_p\_reads - Buffer pool data physical reads monitor element" on page 557  
"pool\_data\_writes - Buffer pool data writes monitor element" on page 559  
"pool\_index\_l\_reads - Buffer pool index logical reads monitor element" on page 564  
"pool\_index\_p\_reads - Buffer pool index physical reads monitor element" on page 566  
"pool\_index\_writes - Buffer pool index writes monitor element" on page 567  
"pool\_no\_victim\_buffer - Buffer pool no victim buffers monitor element" on page 570



“pool\_read\_time - Total buffer pool physical read time monitor element” on page 571

“pool\_temp\_data\_l\_reads - Buffer pool temporary data logical reads monitor element” on page 573

“pool\_temp\_data\_p\_reads - Buffer pool temporary data physical reads monitor element” on page 575

“pool\_temp\_index\_l\_reads - Buffer pool temporary index logical reads monitor element” on page 577

“pool\_temp\_index\_p\_reads - Buffer pool temporary index physical reads monitor element” on page 578

“pool\_write\_time - Total buffer pool physical write time monitor element” on page 584

“tablespace\_name - Table space name monitor element” on page 687

### **event\_thresholdviolations logical data group**

“activate\_timestamp - Activate timestamp monitor element” on page 308

“activity\_collected - Activity collected monitor element” on page 309

“activity\_id - Activity ID monitor element” on page 309

“agent\_id - Application handle (agent ID) monitor element” on page 313

“appl\_id - Application ID” on page 325

“coord\_partition\_num - Coordinator partition number monitor element” on page 383

“destination\_service\_class\_id - Destination service class ID monitor element” on page 402

“source\_service\_class\_id - Source service class ID monitor element” on page 648

“threshold\_action - Threshold action monitor element” on page 704

“threshold\_maxvalue - Threshold maximum value monitor element” on page 705

“threshold\_predicate - Threshold predicate monitor element” on page 706

“threshold\_queuesize - Threshold queue size monitor element” on page 706

“thresholdid - Threshold ID monitor element” on page 706

“time\_of\_violation - Time of violation monitor element” on page 707

“uow\_id - Unit of work ID monitor element” on page 734

### **event\_wlstats logical data group**

“act\_cpu\_time\_top - Activity CPU time top monitor element” on page 304

“act\_rows\_read\_top - Activity rows read top monitor element” on page 307

“concurrent\_wlo\_act\_top - Concurrent WLO activity top monitor element” on page 364

“concurrent\_wlo\_top - Concurrent workload occurrences top monitor element” on page 364

“coord\_act\_aborted\_total - Coordinator activities aborted total monitor element” on page 376

“coord\_act\_completed\_total - Coordinator activities completed total monitor element” on page 376

“coord\_act\_est\_cost\_avg - Coordinator activity estimated cost average monitor element” on page 377

“coord\_act\_exec\_time\_avg - Coordinator activities execution time average monitor element” on page 378



"coord\_act\_interarrival\_time\_avg - Coordinator activity arrival time average monitor element" on page 378  
 "coord\_act\_lifetime\_avg - Coordinator activity lifetime average monitor element" on page 379  
 "coord\_act\_lifetime\_top - Coordinator activity lifetime top monitor element" on page 380  
 "coord\_act\_queue\_time\_avg - Coordinator activity queue time average monitor element" on page 381  
 "coord\_act\_rejected\_total - Coordinator activities rejected total monitor element" on page 382  
 "cost\_estimate\_top - Cost estimate top monitor element" on page 384  
 details\_xml (This XML document contains all the monitor elements reported in the DETAILS column of the MON\_GET\_WORKLOAD\_DETAILS table function output.)  
 "last\_wlm\_reset - Time of last reset monitor element" on page 475  
 "lock\_wait\_time\_top - Lock wait time top monitor element" on page 492  
 "rows\_returned\_top - Actual rows returned top monitor element" on page 627  
 "statistics\_timestamp - Statistics timestamp monitor element" on page 658  
 "uow\_total\_time\_top - UOW total time top monitor element" on page 738  
 "temp\_tablespace\_top - Temporary table space top monitor element" on page 703  
 "wlo\_completed\_total - Workload occurrences completed total monitor element" on page 744  
 "workload\_id - Workload ID monitor element" on page 745  
 "workload\_name - Workload name monitor element" on page 746

### **event\_wcstats logical data group**

"act\_cpu\_time\_top - Activity CPU time top monitor element" on page 304  
 "act\_rows\_read\_top - Activity rows read top monitor element" on page 307  
 "act\_total - Activities total monitor element" on page 307  
 "coord\_act\_est\_cost\_avg - Coordinator activity estimated cost average monitor element" on page 377  
 "coord\_act\_exec\_time\_avg - Coordinator activities execution time average monitor element" on page 378  
 "coord\_act\_interarrival\_time\_avg - Coordinator activity arrival time average monitor element" on page 378  
 "coord\_act\_lifetime\_avg - Coordinator activity lifetime average monitor element" on page 379  
 "coord\_act\_lifetime\_top - Coordinator activity lifetime top monitor element" on page 380  
 "coord\_act\_queue\_time\_avg - Coordinator activity queue time average monitor element" on page 381  
 "cost\_estimate\_top - Cost estimate top monitor element" on page 384  
 "last\_wlm\_reset - Time of last reset monitor element" on page 475  
 "rows\_returned\_top - Actual rows returned top monitor element" on page 627  
 "statistics\_timestamp - Statistics timestamp monitor element" on page 658  
 "temp\_tablespace\_top - Temporary table space top monitor element" on page 703

"work\_action\_set\_id - Work action set ID monitor element" on page 744  
"work\_action\_set\_name - Work action set name monitor element" on page 745  
"work\_class\_id - Work class ID monitor element" on page 745  
"work\_class\_name - Work class name monitor element" on page 745

### **event\_xact logical data group**

"agent\_id - Application handle (agent ID) monitor element" on page 313  
"appl\_id - Application ID" on page 325  
"lock\_escals - Number of lock escalations monitor element" on page 480  
"lock\_wait\_time - Time waited on locks monitor element" on page 490  
"locks\_held\_top - Maximum Number of Locks Held" on page 494  
"partial\_record - Partial Record monitor element" on page 539  
"prev\_uow\_stop\_time - Previous Unit of Work Completion Timestamp" on page 596  
"rows\_read - Rows read monitor element" on page 624  
"rows\_written - Rows Written" on page 629  
"sequence\_no - Sequence number monitor element" on page 635  
"system\_cpu\_time - System CPU Time" on page 677  
"tpmon\_acc\_str - TP monitor client accounting string monitor element" on page 726  
"tpmon\_client\_app - TP monitor client application name monitor element" on page 726  
"tpmon\_client\_userid - TP monitor client user ID monitor element" on page 727  
"tpmon\_client\_wkstn - TP monitor client workstation name monitor element" on page 727  
"uow\_log\_space\_used - Unit of Work Log Space Used" on page 735  
"uow\_start\_time - Unit of Work Start Timestamp" on page 736  
"uow\_status - Unit of Work Status" on page 737  
"uow\_stop\_time - Unit of work stop timestamp monitor element" on page 737  
"user\_cpu\_time - User CPU Time" on page 739  
"x\_lock\_escals - Exclusive Lock Escalations" on page 748

### **lock logical data group**

"data\_partition\_id - Data partition identifier monitor element" on page 390  
"lock\_attributes - Lock attributes monitor element" on page 477  
"lock\_count - Lock count monitor element" on page 478  
"lock\_current\_mode - Original Lock Mode Before Conversion" on page 479  
"lock\_escalation - Lock escalation monitor element" on page 480  
"lock\_hold\_count - Lock hold count monitor element" on page 482  
"lock\_mode - Lock mode monitor element" on page 483  
"lock\_name - Lock name monitor element" on page 484  
"lock\_object\_name - Lock Object Name" on page 485  
"lock\_object\_type - Lock object type waited on monitor element" on page 486  
"lock\_release\_flags - Lock release flags monitor element" on page 486  
"lock\_status - Lock status monitor element" on page 487  
"node\_number - Node Number" on page 517

“table\_file\_id - Table file ID monitor element” on page 678

“table\_name - Table name monitor element” on page 678

“table\_schema - Table schema name monitor element” on page 679

“tablespace\_name - Table space name monitor element” on page 687

### sqlca logical data group

sqlcabc

sqlcaid

sqlcode

sqlerrd

sqlerrmc

sqlerrml

sqlerrp

sqlstate

sqlwarn

---

## Logical data groups affected by COLLECT ACTIVITY DATA settings

The following table shows what logical data groups are collected when different COLLECT ACTIVITY DATA options are specified all types of WLM objects, including Service Subclass, Workload, Work Class (via a Work Action), and Threshold.

Table 76. COLLECT ACTIVITY DATA settings

Setting for COLLECT ACTIVITY DATA	Logical data groups collected
NONE	none
WITHOUT DETAILS	event_activity
WITH DETAILS	event_activity event_activitystmt
WITH DETAILS AND VALUES	event_activity event_activitystmt event_activityvals

---

## Chapter 15. Database system monitor elements

A description of the data collected by the monitor element.

The monitor elements returned by the system monitor fall into the following categories:

- **Identification** for the database manager, an application, or a database connection being monitored.
- Data primarily intended to help you to **configure** the system.
- Database **activity** at various levels including database, application, table, or statement. This information can be used for activity monitoring, problem determination, and performance analysis. It can also be used for configuration.
- Information on **DB2 Connect** applications. Including information on DCS applications running at the gateway, SQL statements being executed, and database connections.
- Information on **Federated Database Systems**. This includes information about the total access to a data source by applications running in a DB2 federated system and information about access to a data source by a given application running in a federated server instance.

Monitor elements are described in a standard format as follows:

### Element identifier

The name of the element. If parsing the data stream directly, the element identifier is uppercase and prefixed with SQLM\_ELM\_.

### Element type

The type of information the monitor element returns. For example, the db2start\_time monitor element returns a timestamp.

### Snapshot monitoring information

If a monitor element returns snapshot monitoring information, a table with the following fields is shown.

- *Snapshot level*: The level of information that can be captured by the snapshot monitor. For example, the appl\_status monitor element returns information at the Application level and at the Lock level.
- *Logical data grouping*: The logical data group where captured snapshot information is returned. If parsing the data stream directly, the logical data group identifier is uppercase and prefixed with SQLM\_ELM\_. For example, the appl\_status monitor element returns information for the appl\_id\_info grouping and for the appl\_lock\_list grouping.
- *Monitor switch*: The system monitor switch that must be set to obtain this information. If the switch is Basic, data will always be collected for the monitor element.

### Event monitoring information

If a monitor element is collected by event monitors, a table with the following fields is shown.

- *Event type*: The level of information that can be collected by the event monitor. The event monitor must be created with this event type to collect this information. For example, the appl\_status monitor element is collected for CONNECTIONS event monitors.

- *Logical data grouping*: The logical data group where captured event information is returned. If parsing the data stream directly, the logical data group identifier is uppercase and prefixed with SQLM\_ELM\_. For example, the appl\_status monitor element returns information for the event\_conn grouping.
- *Monitor switch*: The system monitor switch that must be set to obtain this information. For event monitors, the TIMESTAMP switch is the only monitor switch that can restrict the collection of event data. If there is a dash shown for this field, data will always be collected for the monitor element.

**Usage** Information on how you can use the information collected by the monitor element when monitoring your database system.

---

## acc\_curs\_blk - Accepted Block Cursor Requests

The number of times that a request for an I/O block was accepted.

**Element identifier**  
acc\_curs\_blk

**Element type**  
counter

*Table 77. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

*Table 78. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

**Usage** You can use this element in conjunction with *rej\_curs\_blk* to calculate the percentage of blocking requests that are accepted, rejected, or both.

See *rej\_curs\_blk* for suggestions on how to use this information to tune your configuration parameters.

---

## act\_aborted\_total - Total aborted activities monitor element

The total number of coordinator activities at any nesting level that completed with errors. For service classes, if an activity is remapped to a different service subclass with a REMAP ACTIVITY action before it aborts, then this activity counts only towards the total of the subclass it aborts in.

*Table 79. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 79. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 80. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Use this element to understand if activities on the system are completing successfully. Activities may be aborted due to cancellation, errors, or reactive thresholds.

## act\_completed\_total - Total completed activities monitor element

The total number of coordinator activities at any nesting level that completed successfully. For service classes, if an activity is remapped to a different subclass with a REMAP ACTIVITY action before it completes, then this activity counts only towards the total of the subclass it completes in.

Table 81. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 81. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 82. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Use this element to determine the throughput of activities in the system.

## act\_cpu\_time\_top – Activity CPU time top monitor element

The high watermark for processor time used by activities at all nesting levels in a service class, workload, or work class.

The monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service class or workload in which the activity runs is set to NONE. Activities contribute towards this high watermark only when request metrics are enabled.

For service classes, when you remap activities between service subclasses with a REMAP ACTIVITY action, only the act\_cpu\_time\_top high watermark of the service subclass where an activity completes is updated, provided that a new high watermark is reached. The act\_cpu\_time\_top high watermarks of other service subclasses an activity is mapped to but does not complete in are unaffected.

Table 83. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wcstats	-

Table 83. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wlstats	-

## Usage

Use this element to determine the highest amount of processor time used by an activity on a partition for a service class, workload, or work class during the time interval collected.

---

## act\_exec\_time - Activity execution time monitor element

Time spent executing at this partition, in microseconds. For cursors, the execution time is the combined time for the open, the fetches, and the close. The time when the cursor is idle is not counted towards execution time. For routines, execution time is the start to end of routine invocation. The lifetimes of any cursors left open by routine (to return a result set) after the routine finishes are not counted towards the routine execution time. For all other activities, execution time is the difference between start time and stop time. In all cases, execution time does not include time spent initializing or queued.

Table 84. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

## Usage

This element can be used alone to know the elapsed time spent executing the activity by DB2 on each partition. This element can also be used together with **time\_started** and **time\_completed** monitor elements on the coordinator partition to compute the idle time for cursor activities. You can use the following formula:

Cursor idle time = (time\_completed - time\_started) - act\_exec\_time

---

## act\_rejected\_total - Total rejected activities monitor element

The total number of coordinator activities at any nesting level that were rejected instead of being allowed to execute.

Table 85. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE



Table 85. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 86. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Use this element to help determine whether predictive thresholds and work actions that prevent execution are effective and whether they are too restrictive.

---

## act\_remapped\_in – Activities remapped in monitor element

Count of the number of activities to be remapped into this service subclass since the last reset.

Table 87. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-

## Usage

Use this count to determine whether the remapping of activities into the service subclass is occurring as desired.

---

## act\_remapped\_out – Activities remapped out monitor element

Count of the number of activities to be remapped out of this service subclass since the last reset.

Table 88. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-

## Usage

Use this count to determine whether the remapping of activities out of the service subclass is occurring as desired.

---

## act\_rows\_read\_top – Activity rows read top monitor element

The high watermark for the number of rows read by activities at all nesting levels in a service class, workload, or work class.

The monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service class or workload in which the activity runs is set to NONE. Activities contribute towards this high watermark only when request metrics are enabled.

For service classes, when you remap activities between service subclasses with a REMAP ACTIVITY action only the act\_rows\_read\_top high watermark of the service subclass where an activity completes is updated, provided that a new high watermark is reached. The act\_rows\_read\_top high watermarks of service subclasses an activity is mapped to but does not complete in are unaffected.

*Table 89. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wcstats	-
Statistics	event_wlstats	-

## Usage

Use this element to determine the highest number of rows read by an activity on a partition for a service class, workload, or work class during the time interval collected.

---

## act\_total - Activities total monitor element

Total number of activities at any nesting level that had work actions corresponding to the specified work class applied to them since the last reset.

*Table 90. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wcstats	-

## Usage

Every time an activity has one or more work actions associated with a work class applied to it, a counter for the work class is updated. This counter is exposed using the act\_total monitor element. The counter can be used to judge the effectiveness of the work action set (for example, how many activities have a actions applied). It can also be used to understand the different types of activities on the system.

---

## activate\_timestamp - Activate timestamp monitor element

The time when an event monitor was activated.

Table 91. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activity	event_activity	-
Activity	event_activitystmt	-
Activity	event_activityvals	-
Threshold Violations	event_thresholdviolations	-

### Usage

Use this element to correlate information returned by the above event types.

---

## active\_hash\_joins - Active hash joins

The total number of hash joins that are currently running and consuming memory.

Table 92. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	-

---

## active\_olap\_funcs - Active OLAP Functions monitor element

The total number of OLAP functions that are currently running and consuming sort heap memory.

Table 93. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	-

For snapshot monitoring, this counter can be reset.

---

## active\_sorts - Active Sorts

The number of sorts in the database that currently have a sort heap allocated.

Table 94. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

**Usage** Use this value in conjunction with *sort\_heap\_allocated* to determine the average sort heap space used by each sort. If the *sortheap* configuration parameter is substantially larger than the average sort heap used, you may be able to lower the value of this parameter.

This value includes heaps for sorts of temporary tables that were created during relational operations.

---

## activity\_collected - Activity collected monitor element

This element indicates whether or not activity event monitor records are to be collected for a violated threshold.

Table 95. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Threshold violations	event_thresholdviolations	-

### Usage

Use this element to determine whether to expect an activity event for the activity that violated the threshold to be written to the activity event monitor.

When an activity finishes or aborts and the activity event monitor is active at the time, if the value of this monitor element is 'Y', the activity that violated this threshold will be collected. If the value of this monitor element is 'N', it will not be collected.

---

## activity\_id - Activity ID monitor element

Counter which uniquely identifies an activity for an application within a given unit of work.

Table 96. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 97. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Activities	event_activity	-
Activities	event_activitystmt	-
Activities	event_activityvals	-
Threshold violations	event_thresholdviolations	-

### Usage

Use this element in conjunction with other activity history elements for analysis of the behavior of an activity.

To uniquely identify an activity outside its unit of work, use the combination of **activity\_id** and **uow\_id** plus one of the following: **appl\_id** or **agent\_id**.

---

## activity\_secondary\_id - Activity secondary ID monitor element

The value for this element is incremented each time an activity record is written for the same activity. For example, if an activity record is written once as a result of having called the WLM\_CAPTURE\_ACTIVITY\_IN\_PROGRESS procedure and a second time when the activity ends, the element would have a value of 0 for the first record and 1 for the second record.

Table 98. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-
Activities	event_activitystmt	-
Activities	event_activityvals	-

### Usage

Use this element with **activity\_id**, **uow\_id**, and **appl\_id** monitor elements to uniquely identify activity records when information about the same activity has been written to the activities event monitor multiple times.

For example, information about an activity would be sent to the activities event monitor twice in the following case:

- the WLM\_CAPTURE\_ACTIVITY\_IN\_PROGRESS stored procedure was used to capture information about the activity while it was running
- information about the activity was collected when the activity completed, because the COLLECT ACTIVITY DATA clause was specified on the service class with which the activity is associated

---

## activity\_state - Activity state monitor element

The current state of the activity.

Table 99. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### Usage

Use this monitor element to understand what the activity is currently doing (for example, is the activity stuck in a queue or waiting for input from the client). Possible values include:

- CANCEL\_PENDING
- EXECUTING
- IDLE
- INITIALIZING
- QP\_CANCEL\_PENDING
- QP\_QUEUED
- QUEUED

- TERMINATING
- UNKNOWN

---

## activity\_type - Activity type monitor element

The type of the activity.

*Table 100. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

*Table 101. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

### Usage

The possible values are:

- LOAD
- READ\_DML
- WRITE\_DML
- DDL
- CALL
- OTHER

---

## activitytotaltime\_threshold\_id - Activity total time threshold ID monitor element

The ID of the ACTIVITYTOTALTIME threshold that was applied to the activity.

*Table 102. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### Usage

Use this element to understand which ACTIVITYTOTALTIME threshold, if any, was applied to the activity.

---

## activitytotaltime\_threshold\_value - Activity total time threshold value monitor element

A timestamp that is computed by adding the ACTIVITYTOTALTIME threshold duration to the activity entry time. If the activity is still executing when this timestamp is reached, the threshold will be violated.

Table 103. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### Usage

Use this element to understand the value of the ACTIVITYTOTALTIME threshold applied to the activity, if any.

---

## activitytotaltime\_threshold\_violated - Activity total time threshold violated monitor element

This monitor element returns 'Yes' to indicate that the activity violated the ACTIVITYTOTALTIME threshold. 'No' indicates that the activity has not yet violated the threshold.

Table 104. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### Usage

Use this element to determine if the activity violated the ACTIVITYTOTALTIME threshold that was applied to the activity.

---

## address - IP address from which the connection was initiated

The IP address from which the activity connection was initiated.

Table 105. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

### Usage

Use this to identify the IP address from which the activity connection was initiated. Secure domain names are shown converted to an IP address.

## agent\_id - Application handle (agent ID) monitor element

A system-wide unique ID for the application. On a single-partitioned database, this identifier consists of a 16-bit counter. On a multi-partitioned database, this identifier consists of the coordinating partition number concatenated with a 16-bit counter. In addition, this identifier is the same on every partition where the application may make a secondary connection.

Table 106. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	Always collected
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 107. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_id_info	Basic
Lock	appl_lock_list	Basic
DCS Application	dcs_appl_info	Basic

Table 108. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-
Connections	event_connheader	-
Statements	event_stmt	-
Statements	event_subsection	-
Deadlocks <sup>1</sup>	event_dlconn	-
Deadlocks with Details <sup>1</sup>	event_detailed_dlconn	-
Threshold violations	event_thresholdviolations	-
Activities	event_activity	-

- <sup>1</sup> This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.



## Usage

The application handle, also known as the agent ID, can be used to uniquely identify an active application.

**Note:** The **agent\_id** monitor element has different behavior depending on your version of DB2. When taking snapshots from DB2 with version SQLM\_DBMON\_VERSION1 or SQLM\_DBMON\_VERSION2 to a DB2 (Version 5 or greater) database, the **agent\_id** returned is not usable as an application identifier, rather it is the **agent\_pid** of the agent serving the application. In these cases an **agent\_id** is still returned for compatibility with earlier releases, but internally the DB2 database server will not recognize the value as an **agent\_id**.

This value can be used as input to GET SNAPSHOT commands that require an agent ID or to the monitor table functions that require an application handle.

When reading event traces, it can be used to match event records with a given application.

It can also be used as input to the FORCE APPLICATION command or API. On multi-node systems this command can be issued from any node where the application has a connection. Its effect is global.

---

## agent\_id\_holding\_lock - Agent ID Holding Lock

The application handle of the agent holding a lock for which this application is waiting. The lock monitor group must be turned on to obtain this information.

*Table 109. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Lock
Lock	appl_lock_list	Lock
Lock	lock_wait	Lock

**Usage** This element can help you determine which applications are in contention for resources.

If this element is 0 (zero) and the application is waiting for a lock, this indicates that the lock is held by an indoubt transaction. You can use either `appl_id_holding_lk` or the command line processor `LIST INDOUBT TRANSACTIONS` command (which displays the application ID of the CICS agent that was processing the transaction when it became indoubt) to determine the indoubt transaction, and then either commit it or roll it back.

Note that more than one application can hold a shared lock on an object for which this application is waiting. See `lock_mode` for information about the type of lock that the application holds. If you are taking an application snapshot, only one of the agent IDs holding a lock on the object will be returned. If you are taking a lock snapshot, all of the agent IDs holding a lock on the object will be identified.

---

## agent\_pid - Engine dispatchable unit (EDU) identifier monitor element

The unique identifier for the engine dispatchable unit (EDU) for the agent. Except on the Linux operating system, the EDU ID is mapped to the thread ID. On the Linux operating system, the EDU ID is a DB2 generated unique identifier.

Table 110. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	agent	Statement

Table 111. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-

### Usage

You can use this element to link database system monitor information to other sources of diagnostic information, such as system traces. You can also use it to monitor how agents working for a database application use system resources.

---

## agent\_status - DCS Application Agents

In a connection concentrator environment, this value shows which applications currently have associated agents.

### Element identifier

agent\_status

### Element type

information

Table 112. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcx_appl_info	Basic

**Usage** Values are:

- SQLM\_AGENT\_ASSOCIATED  
The agent working on behalf of this application is associated with it.
- SQLM\_AGENT\_NOT\_ASSOCIATED  
The agent that was working on behalf of this application is no longer associated with it and is being used by another application. The next time work is done for this application without an associated agent, an agent will be re-associated.

---

## agent\_sys\_cpu\_time - System CPU Time used by Agent

The total *system* CPU time (in seconds and microseconds) used by the database manager agent process.

### Element identifier

agent\_sys\_cpu\_time

### Element type

time

Table 113. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Timestamp

For snapshot monitoring at the application level, this counter can be reset. This counter cannot be reset at other levels.

**Usage** This element along with the other related CPU-time elements can help you understand the level of activity within an application, and may help you identify applications that could benefit from additional tuning.

This element includes CPU time for both SQL and non-SQL statements, as well as CPU time for any unfenced user-defined functions (UDFs)

System CPU represents the time spent in system calls. User CPU represents time spent executing database manager code.

**Note:** If this information is not available for your operating system, this element will be set to 0.

---

## agent\_usr\_cpu\_time - User CPU Time used by Agent

The total CPU time (in seconds and microseconds) used by database manager agent process.

**Element identifier**

agent\_usr\_cpu\_time

**Element type**

time

Table 114. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Timestamp

For snapshot monitoring, this counter can be reset.

**Usage** This element along with the other CPU-time related elements can help you identify applications or queries that consume large amounts of CPU.

This counter includes time spent on both SQL and non-SQL statements, as well as any unfenced user-defined functions (UDFs) or stored procedures executed by the application.

System CPU represents the time spent in system calls. User CPU represents time spent executing database manager code.

**Note:** If this information is not available for your operating system, this element will be returned as 0.

---

## agent\_wait\_time - Agent wait time monitor element

Time spent by an application queued to wait for an agent under concentrator configurations. The value is given in milliseconds.

Table 115. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 116. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

The **agent\_wait\_time** monitor element can be used to help evaluate how efficiently your system is running in a concentrator environment. A high agent wait relative to the **total\_request\_time** monitor element value indicates that requests are spending a lot of time queued waiting for agents, which may be indicative of one or more of the following:

- The **max\_coordagents** configuration parameter has been configured too small for your workload. You may need to increase the value of **max\_coordagents** configuration parameter, or the ratio of **max\_coordagents** configuration parameter to **max\_connections** configuration parameter if you are running with both parameters set to AUTOMATIC, to ensure that enough coordinator agents are available to service your application requests in a timely manner.
- Your workload is not committing frequently enough. For the concentrator to work efficiently, applications should issue commits relatively frequently to ensure that their agents can be freed up to serve requests on other applications.

If your applications do not do frequent commits you may need to configure a proportionally higher number of coordinator agents to reduce the time spent waiting for agents to become available.

## agent\_waits\_total - Total agent waits monitor element

Number of times an application had to wait for an agent to be assigned under concentrator configurations.

Table 117. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 118. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

### Usage

Use this element in conjunction with the **agent\_wait\_time** monitor element to determine the average amount of time an application request spends waiting for an agent in a concentrator environment.

---

## agents\_created\_empty\_pool - Agents Created Due to Empty Agent Pool

The number of agents created because the agent pool was empty. It includes the number of agents started at DB2 start up (*num\_initagents*).

Table 119. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

**Usage** In conjunction with `agents_from_pool`, you can calculate the ratio of Agents Created Due to Empty Agent Pool / Agents Assigned From Pool

See `agents_from_pool` for information on using this element.

---

## agents\_from\_pool - Agents Assigned From Pool

The number of agents assigned from the agent pool.

Table 120. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

### Usage

This element can be used with the `agents_created_empty_pool` monitor element to determine how often an agent must be created because the pool is empty.

The following ratio

Agents Created Due to Empty Agent Pool / Agents Assigned From Pool

can be used to help set an appropriate value for the `num_poolagents` configuration parameter.

For most users, the default value of 100 with AUTOMATIC will ensure optimal performance.

This ratio may fluctuate somewhat with the workload. At times of low activity on the system, additional agent creation and termination may occur. At times of high activity on the system, more agent reuse will occur. A low ratio indicates that there is a high amount of agent reuse, which is expected on systems with high activity. A high ratio indicates a higher amount of agent creation than reuse is occurring. If this is a concern, increase the value for the `num_poolagents` configuration parameter to lower the ratio. However, this will cause additional resources consumption on the system.

---

## agents\_registered - Agents Registered

The number of agents registered in the database manager instance that is being monitored (coordinator agents and subagents).

Table 121. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

## Usage

Use this element to help evaluate your settings for the **max\_coordagents** and **max\_connections** configuration parameters, as well as the intraquery parallelism settings.

---

## agents\_registered\_top - Maximum Number of Agents Registered

The maximum number of agents that the database manager has ever registered, at the same time, since it was started (coordinator agents and subagents).

Table 122. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

## Usage

You may use this element to help you evaluate your settings for the **max\_coordagents** and **max\_connections** configuration parameters, as well as the intraquery parallelism settings.

The number of agents registered at the time the snapshot was taken is recorded by the **agents\_registered** monitor element.

---

## agents\_stolen - Stolen Agents

At the database manager snapshot level, this monitor element represents the number of idle agents associated with an application which get reassigned to work on a different application. At the application snapshot level, this monitor element represents the number of idle agents associated with a different application which get reassigned to work on this application.

Table 123. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

## Usage

The **num\_poolagents** configuration parameter is set to **AUTOMATIC** by default. This means that DB2 automatically manages the pooling of idle agents, which includes assigning work to idle agents associated with another application.

---

## agents\_top - Number of Agents Created

At the application level, this is the maximum number of agents that were used when executing the statement. At the database level, it is the maximum number of agents for all applications.

**Element identifier**

agents\_top

**Element type**

watermark

*Table 124. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Statement
Application	stmt	Statement

**Usage** An indicator how well intra-query parallelism was realized.

---

## agents\_waiting\_on\_token - Agents Waiting for a Token

The number of agents waiting for a token so they can execute a transaction in the database manager.

**Note:** The **agents\_waiting\_on\_token** monitor element is deprecated starting with DB2 Version 9.5. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

*Table 125. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

### Usage

You can use this element to help evaluate your setting for the **maxcagents** configuration parameter.

Each application has a dedicated coordinator agent to process database requests within the database manager. Each agent has to get a token before it can execute a transaction. The maximum number of agents that can execute database manager transactions is limited by the configuration parameter **maxcagents**.

---

## agents\_waiting\_top - Maximum Number of Agents Waiting monitor element

The maximum number of agents that have ever been waiting for a token, at the same time, since the database manager was started.

**Note:** The **agents\_waiting\_top** monitor element is deprecated starting with DB2 Version 9.5. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.



Table 126. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

## Usage

Use this element to help you evaluate your setting of the **maxcagents** configuration parameter.

The number of agents waiting for a token at the time the snapshot was taken is recorded by the **agents\_waiting\_on\_token** monitor element.

If the **maxcagents** parameter is set to its default value (-1), no agents should wait for a token and the value of this monitor element should be zero.

---

## agg\_temp\_tablespace\_top - Aggregate temporary table space top monitor element

The high watermark in KB for the aggregate temporary table space usage of DML activities at all nesting levels in a service class. The aggregate is computed by summing the temporary table space usage across all activities in the service subclass, and this high watermark represents the highest value reached by this aggregate since the last reset. The monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service class is set to NONE. An AGGSQLTEMPSPACE threshold must be defined and enabled for at least one service subclass in the same superclass as the subclass to which this record belongs, otherwise a value of 0 is returned.

Table 127. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-

## Usage

Use this element to determine the highest aggregate DML activity system temporary table space usage reached on a partition for a service subclass in the time interval collected.

---

## aggsqltempespace\_threshold\_id - Aggregate SQL temporary space threshold ID monitor element

The numeric ID of the AGGSQLTEMPSPACE threshold that was applied to the activity.

Table 128. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

Use this element to understand which AGGSQLTEMPSPACE threshold, if any, was applied to the activity.

---

### aggsqltempespace\_threshold\_value - AggSQL temporary space threshold value monitor element

The upper bound of the AGGSQLTEMPSPACE threshold that was applied to the activity.

*Table 129. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

Use this element to understand the value of the AGGSQLTEMPSPACE threshold applied to the activity, if any.

---

### aggsqltempespace\_threshold\_violated - AggSQL temporary space threshold violated monitor element

The optional monitor element when set to 'Yes' indicates that the activity violated the AGGSQLTEMPSPACE threshold that was applied to it. 'No' indicates that the activity has not yet violated the threshold.

*Table 130. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

Use this element to determine if the activity violated the AGGSQLTEMPSPACE threshold that was applied to the activity.

---

### app\_rqsts\_completed\_total - Total application requests completed monitor element

Total number of external (application) requests executed by the coordinator. For service subclasses, this monitor element is updated only for the subclass where the application request completes.

*Table 131. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE

Table 131. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 132. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Use this monitor element to understand how many requests are being submitted into the system from applications.

---

## appl\_con\_time - Connection Request Start Timestamp

The date and time that an application started a connection request.

### Element identifier

appl\_con\_time

### Element type

timestamp

Table 133. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Timestamp

**Usage** Use this element to determine when the application started its connection request to the database.

---

## appl\_id - Application ID

This identifier is generated when the application connects to the database at the database manager or when DB2 Connect receives a request to connect to a DRDA® database.

*Table 134. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_id_info	Basic
DCS Application	dcs_appl_info	Basic
Lock	appl_lock_list	Basic

*Table 135. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-
Connection	event_conn	-
Connections	event_connheader	-
Statements	event_stmt	-
Transactions <sup>1</sup>	event_xact	-
Deadlocks <sup>2</sup>	event_dlconn	-
Deadlocks with Details <sup>2</sup>	event_detailed_dlconn	-
Activities	event_activitystmt	-
Activities	event_activity	-
Activities	event_activityvals	-
Threshold violations	event_thresholdviolations	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR UNIT OF WORK statement to monitor transaction events.
- 2 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

### Usage

This ID is known on both the client and server, so you can use it to correlate the client and server parts of the application. For DB2 Connect applications, you will also need to use **outbound\_appl\_id** monitor element to correlate the client and server parts of the application.

This identifier is unique across the network. There are different formats for the application ID, which are dependent on the communication protocol between the client and the server machine on which the database manager, DB2 Connect, or both are running. Each of the formats consists of three parts separated by periods.

## 1. TCP/IP

### Format

IPAddr.Port.Application instance

### IPv4

#### Example

G91A3955.F33A.02DD18143340

#### Details

In IPv4, a TCP/IP-generated application ID is composed of three sections. The first section contains the IP address. It is represented as a 32-bit number displayed as a maximum of 8 hexadecimal characters. The second section contains the port number, which is represented as 4 hexadecimal characters. The third section contains a unique identifier for the instance of this application.

**Note:** When the hexadecimal versions of the IP address or port number begin with 0-9, they are changed to G-P respectively. For example, "0" is mapped to "G", "1" is mapped to "H", and so on.

The IP address, AC10150C.NA04.006D07064947 is interpreted as follows:

- The IP address remains AC10150C, which translates to 172.16.21.12.
- The port number is NA04. The first character is "N", which maps to "7". Therefore, the hexadecimal form of the port number is 7A04, which translates to 31236 in decimal form.

### IPv6

#### Example

1111:2222:3333:4444:5555:6666:  
7777:8888.65535.0123456789AB

#### Details

In IPv6, a TCP/IP-generated application ID is composed of three sections. The first section contains the IP address which is a 39 byte readable address of the form a:b:c:d:e:f:g:h, where each of a-h is 4 hexadecimal digits. The second section is a readable 5-byte port number. The third section is a unique timestamp identifier for the instance of this application.

## 2. Local Applications

### Format

\*LOCAL.DB2 instance.Application instance

### Example

\*LOCAL.DB2INST1.930131235945

### Details

The application ID generated for a local application is made up by concatenating the string \*LOCAL, the name of the DB2 instance, and a unique identifier for the instance of this application.

For multiple database partition instances, LOCAL is replaced with Nx, where x is the partition number from which the client connected to the database. For example, \*N2.DB2INST1.0B5A12222841.

Use the **client\_protocol** monitor element to determine which communications protocol the connection is using and, as a result, the format of the **appl\_id** monitor element.

---

## appl\_id\_holding\_lk - Application ID Holding Lock

The application ID of the application that is holding a lock on the object that this application is waiting to obtain.

**Element identifier**  
appl\_id\_holding\_lk

**Element type**  
information

*Table 136. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Lock
Lock	appl_lock_list	Lock
Lock	lock_wait	Lock

*Table 137. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	event_dlconn	-
Deadlocks with Details	event_detailed_dlconn	-

**Usage** This element can help you determine which applications are in contention for resources. Specifically, it can help you identify the application handle (agent ID) and table ID that are holding the lock. Note that you may use the LIST APPLICATIONS command to obtain information to relate the application ID with an agent ID. However, it is a good idea to collect this type of information when you take the snapshot, as it could be unavailable if the application ends before you run the LIST APPLICATIONS command.

Note that more than one application can hold a shared lock on an object for which this application is waiting to obtain a lock. See lock\_mode for information about the type of lock that the application holds. If you are taking an application snapshot, only one of the application IDs holding a lock on the object will be returned. If you are taking a lock snapshot, all of the application IDs holding a lock on the object will be returned.

---

## appl\_id\_oldest\_xact - Application with Oldest Transaction

The application ID (which corresponds to the *agent\_id* value from the application snapshot) of the application that has the oldest transaction.

**Element identifier**  
appl\_id\_oldest\_xact

**Element type**  
information

*Table 138. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

**Usage** This element can help you determine which application has the oldest active transaction. This application can be forced to free up log space. If it is taking up a great deal of log space, you should examine the application to determine if it can be modified to commit more frequently.

There are times when there is not a transaction holding up logging, or the oldest transaction does not have an application ID (for example, indoubt transaction or inactive transaction). In these cases, this application's ID is not returned in the data stream.

---

## appl\_idle\_time - Application Idle Time

Number of seconds since an application has issued any requests to the server. This includes applications that have not terminated a transaction, for example not issued a commit or rollback.

**Element identifier**

appl\_idle\_time

**Element type**

information

*Table 139. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Statement
DCS Application	dcs_appl	Statement

**Usage** This information can be used to implement applications that force users that have been idle for a specified number of seconds.

---

## appl\_name - Application name monitor element

The name of the application running at the client, as known to the database or DB2 Connect server.

*Table 140. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_id_info	Basic
Lock	appl_lock_list	Basic
DCS Application	dcs_appl_info	Basic

*Table 141. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-
Connections	event_connheader	-
Activities	event_activity	-

**Usage**

This element can be used with **appl\_id** to relate data items with your application.

In a client-server environment, this name is passed from the client to the server when establishing the database connection. A CLI application can set the `SQL_ATTR_INFO_PROGRAMNAME` attribute with a call to `SQLSetConnectAttr`. When `SQL_ATTR_INFO_PROGRAMNAME` is set before the connection to the server is established, the value specified overrides the actual client application name and will be the value that is displayed in the `appl_name` monitor element.

In situations where the client application code page is different from the code page under which the database system monitor is running, you can use `codepage_id` to help translate `appl_name`.

---

## appl\_priority - Application Agent Priority

The priority of the agents working for this application.

**Element identifier**

`appl_priority`

**Element type**

information

*Table 142. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

*Table 143. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

**Usage** You can use this element to check if applications are running with the expected priorities. Application priorities can be set by an administrator. They can be changed by the governor utility (**db2gov**).

The governor is used by DB2 to monitor and change the behavior of applications running against a database. This information is used to schedule applications and balance system resources.

A governor daemon collects statistics about the applications by taking snapshots. It checks these statistics against the rules governing applications running on that database. If the governor detects a rule violation, it takes the appropriate action. These rules and actions were specified by you in the governor configuration file.

If the action associated with a rule is to change an application's priority, the governor changes the priority of the agents in the partition where the violation was detected.

---

## appl\_priority\_type - Application Priority Type

Operating system priority type for the agent working on behalf of the application.

**Element identifier**

`appl_priority_type`

**Element type**

information



Table 144. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

Table 145. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

**Usage** Dynamic priority is recalculated by the operating system based on usage. Static priority does not change.

---

## appl\_section\_inserts - Section Inserts monitor element

Inserts of SQL sections by an application from its shared SQL workspace.

Table 146. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

Table 147. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

### Usage

The working copy of any executable section is stored in a shared SQL workspace. This is a count of when a copy was not available and had to be inserted.

---

## appl\_section\_lookups - Section Lookups

Lookups of SQL sections by an application from its shared SQL workspace.

Table 148. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 149. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

## Usage

Each agent has access to a shared SQL workspace where the working copy of any executable section is kept. This counter indicates how many times the SQL work area was accessed by agents for an application.

---

## appl\_status - Application Status

The current status of the application.

*Table 150. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_id_info	Basic
Lock	appl_lock_list	Basic

*Table 151. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Connection	event_conn	-

## Usage

This element can help you diagnose potential application problems. Values for this field are listed in the following table.

API Constant	Description
SQLM_AUTONOMOUS_WAIT	<b>Autonomous Wait:</b> The application is waiting for an autonomous routine to complete.
SQLM_BACKUP	<b>Backing Up Database:</b> The application is performing a backup of the database.
SQLM_COMMIT_ACT	<b>Commit Active:</b> The unit of work is committing its database changes.
SQLM_COMP	<b>Compiling:</b> The database manager is compiling an SQL statement or precompiling a plan on behalf of the application.
SQLM_CONNECTED	<b>Database Connect Completed:</b> The application has initiated a database connection and the request has completed.
SQLM_CONNECTPEND	<b>Database Connect Pending:</b> The application has initiated a database connection but the request has not yet completed.
SQLM_CREATE_DB	<b>Creating Database:</b> The agent has initiated a request to create a database and that request has not yet completed.

API Constant	Description
SQLM_DECOUPLED	<b>Decoupled from Agent:</b> There are no agents currently associated with the application. This is a normal state. When the Connection Concentrator is enabled, there is no dedicated coordinator agent, so an application can be decoupled on the coordinator partition. In non-concentrator environments, an application cannot be decoupled on the coordinator partition as there will always be a dedicated coordinator agent .
SQLM_DISCONNECTPEND	<b>Database Disconnect Pending:</b> The application has initiated a database disconnect but the command has not yet completed executing. The application may not have explicitly executed the database disconnect command. The database manager will disconnect from a database if the application ends without disconnecting.
SQLM_INTR	<b>Request Interrupted:</b> An interrupt of a request is in progress.
SQLM_IOERROR_WAIT	<b>Wait to Disable Table space:</b> The application has detected an I/O error and is attempting to disable a particular table space. The application has to wait for all other active transactions on the table space to complete before it can disable the table space.
SQLM_LOAD	<b>Data Fast Load:</b> The application is performing a "fast load" of data into the database.
SQLM_LOCKWAIT	<b>Lock Wait:</b> The unit of work is waiting for a lock. After the lock is granted, the status is restored to its previous value.
SQLM_QUIESCE_TABLESPACE	<b>Quiescing a Table space:</b> The application is performing a quiesce table space request.
SQLM_RECOMP	<b>Recompiling:</b> The database manager is recompiling (that is, rebinding) a plan on behalf of the application.
SQLM_REMOTE_RQST	<b>Federated request pending:</b> The application is waiting for results from a federated data source.
SQLM_RESTART	<b>Restarting Database:</b> The application is restarting a database in order to perform crash recovery.
SQLM_RESTORE	<b>Restoring Database:</b> The application is restoring a backup image to the database.
SQLM_ROLLBACK_ACT	<b>Rollback Active:</b> The unit of work is rolling back its database changes.
SQLM_ROLLBACK_TO_SAVEPOINT	<b>Rollback to savepoint:</b> The application is rolling back to a savepoint.
SQLM_TEND	<b>Transaction Ended:</b> The unit of work is part of a global transaction that has ended but has not yet entered the prepared phase of the two-phase commit protocol.
SQLM_THABRT	<b>Transaction Heuristically Rolled Back:</b> The unit of work is part of a global transaction that has been heuristically rolled-back.

API Constant	Description
SQLM_THCOMT	<b>Transaction Heuristically Committed:</b> The unit of work is part of a global transaction that has been heuristically committed.
SQLM_TPREP	<b>Transaction Prepared:</b> The unit of work is part of a global transaction that has entered the prepared phase of the two-phase commit protocol.
SQLM_UNLOAD	<b>Data Fast Unload:</b> The application is performing a “fast unload” of data from the database.
SQLM_UOWEXEC	<b>Unit of Work Executing:</b> The database manager is executing requests on behalf of the unit of work.
SQLM_UOWWAIT	<b>Unit of Work waiting:</b> The database manager is waiting on behalf of the unit of work in the application. This status typically means that the system is executing in the application’s code.
SQLM_WAITFOR_REMOTE	<b>Pending remote request:</b> The application is waiting for a response from a remote partition in a partitioned database instance.

## application\_handle - Application handle monitor element

A system-wide unique ID for the application. On a single-partitioned database, this identifier consists of a 16-bit counter. On a multi-partitioned database, this identifier consists of the coordinating partition number concatenated with a 16-bit counter. In addition, this identifier is the same on every partition where the application may make a secondary connection.

Table 152. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	Always collected
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 153. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_id_info	Basic
Lock	appl_lock_list	Basic
DCS Application	dcs_appl_info	Basic

Table 154. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-
Connections	event_connheader	-
Statements	event_stmt	-
Statements	event_subsection	-
Deadlocks <sup>1</sup>	event_dlconn	-
Deadlocks with Details <sup>1</sup>	event_detailed_dlconn	-
Threshold violations	event_thresholdviolations	-
Activities	event_activity	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

## Usage

This monitor element is an alias of the **agent\_id** monitor element.

---

## appls\_cur\_cons - Applications Connected Currently

Indicates the number of applications that are currently connected to the database.

Table 155. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Lock	db_lock_list	Basic

**Usage** You may use this element to help you understand the level of activity within a database and the amount of system resource being used.

It can help you adjust the setting of the *maxappls* and *max\_coordagents* configuration parameters. For example, its value is always the same as *maxappls*, you may want to increase the value of *maxappls*. See the *rem\_cons\_in* and the *local\_cons* monitor elements for more information.

---

## appls\_in\_db2 - Applications Executing in the Database Currently

Indicates the number of applications that are currently connected to the database, and for which the database manager is currently processing a request.

Table 156. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

---

## arm\_correlator - Application response measurement correlator monitor element

Identifier of a transaction in the Application Response Measurement (ARM) standard.

Table 157. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

### Usage

This element can be used to link an activity collected by the activities event monitor to the applications associated with the activity, if such applications also support the Application Response Measurement (ARM) standard.

---

## associated\_agents\_top - Maximum Number of Associated Agents

The maximum number of subagents associated with this application.

Table 158. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

---

## async\_runstats – Total number of asynchronous RUNSTATS requests monitor element

The total number of successful asynchronous RUNSTATS activities performed by real-time statistics gathering for all the applications in the database. Values reported by all the database partitions are aggregated together.

Table 159. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Statement

For snapshot monitoring, this counter can be reset.

Table 160. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

### Usage

Use this element to determine how many successful asynchronous RUNSTATS activities have been performed by real-time statistics gathering. This value changes frequently. In order to get a better view of the system usage, take a snapshot at specific intervals over an extended period of time. When used in conjunction with **sync\_runstats** and **stats\_fabrications** monitor elements, this element can help you to track the different types of statistics collection activities related to real-time statistics gathering and analyze their performance impact.

---

## audit\_events\_total - Total audit events monitor element

The total number of audit events generated.

Table 161. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

Table 162. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

---

## audit\_file\_write\_wait\_time - Audit file write wait time monitor element

Time spent waiting to write an audit record. The value is given in milliseconds.

Table 163. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 163. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

Table 164. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Use this monitor element to determine the amount of time an agent spends waiting to open and write an audit event synchronously to disk.

In a typical scenario, only one agent attempts to open the audit log file at a time, as the other agents wait for access to the audit common subsystem before opening the file. Therefore, the wait time usually represents the time spent waiting to write the file to disk by the operating system. Audit utilities might lock the audit log file during execution, which causes a longer than normal wait time for agents to open and write to the audit log file. If asynchronous auditing is enabled, audit events that are larger than the asynchronous audit buffer are written directly to disk, instead of to the buffer, and contribute to the wait time.

Outside of the special audit utility scenario, the wait time depends on the speed of the disks and how quickly the operating system can write the data to them. In order to reduce this wait time for a given application and audit configuration, you might tune the operating system or use faster disks.

---

## audit\_file\_writes\_total - Total audit files written monitor element

The total number of times an agent has had to wait to write an audit event directly to disk.

Table 165. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE



Table 165. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

Table 166. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Use this monitor element in conjunction with the **audit\_file\_write\_wait\_time** monitor element to determine the average time an application request spends waiting to open and write an audit event synchronously to disk.

## audit\_subsystem\_wait\_time - Audit subsystem wait time monitor element

Time spent waiting for space in audit buffer. Waiting occurs when audit buffer is full and agent must wait for audit daemon to write buffer to disk. The value is given in milliseconds.

Table 167. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 167. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

Table 168. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Use this monitor element to determine the amount of time an agent spends waiting to access the audit common subsystem, while the audit common subsystem is busy handling events for other agents.

Certain common portions of the audit subsystem can only be accessed by a single agent at a time. The value of this monitor element indicates the amount of time that an agent must wait to access the audit common subsystem. This includes time spent by an agent that has filled the current asynchronous buffer waiting for the audit daemon to finish writing out a previous asynchronous buffer to disk. Other agents that are waiting while writing to the audit log file or waiting to make a request of the audit daemon have also accessed the audit common subsystem and wait times there will be reflected in this value.

To reduce this wait time, you might change the value of the **audit\_buf\_sz** configuration parameter if asynchronous auditing is in use. You can increase the value of the **audit\_buf\_sz** configuration parameter until further increases no longer show any reductions in the audit common subsystem wait time. At this point, the asynchronous buffers are large enough such that the daemon is able to write one full buffer to disk before the next buffer is full, and then the daemon is no longer a bottleneck. If the value of the **audit\_buf\_sz** configuration parameter must be increased to such an extent that too many audit records could be lost if a system failure were to occur, then you might reduce the wait time by tuning the operating system or using faster disks. If further reduction in the wait time is necessary, then use audit policies to reduce the number of audit events generated.

---

## audit\_subsystem\_waits\_total - Total audit subsystem waits monitor element

Number of times audit has waited for a buffer write.

*Table 169. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

*Table 170. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

### Usage

Use this monitor element to determine the total number of times an agent has had to access the audit common subsystem. The generation of one audit event may need to access the audit common subsystem none, one, or more times to record the event. Use the **audit\_events\_total** monitor element to determine the exact number of audit events generated.

---

## auth\_id - Authorization ID

The authorization ID of the user who invoked the application that is being monitored. On a DB2 Connect gateway node, this is the user's authorization ID on the host.

Table 171. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_id_info	Basic
Lock	appl_lock_list	Basic
DCS Application	dcs_appl_info	Basic

Table 172. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-
Connections	event_connheader	-

## Usage

In an explicit trusted connection, the **auth\_id** value does not change immediately when you switch users. Rather, the **auth\_id** is updated the first time you access the database after switching users. This is because the switch user operation is always chained to the subsequent operation.

You can use this element to determine who invoked the application.

---

## authority\_bitmap - User authorization level monitor element

The authorities granted to the user and to the groups to which the user belongs. These include authorities granted to roles that are granted to the user and to the groups to which the user belongs. Authorities granted to a user or to roles granted to the user are considered user authorities. Authorities granted to a group to which the user belongs or to roles granted to the group to which the user belongs are considered group authorities.

Table 173. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic
Application	appl_info	Basic

Table 174. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

## Usage

The **authority\_bitmap** monitor element has the format of an array. Each array element is a single character that represents whether or not the user ID has been granted a specific authority and how the user has received that authority.

Individual array elements are indexed through an index value defined in the `sql.h` file. The value of an index in the **authority\_bitmap** array is called an *authority index*. For example, `SQL_DBAUTH_SYSADM` is the index to determine if the user has `SYSADM` authority.

The value of one element in the `authority_bitmap` array identified by an authority index represents whether the authority is held by an authorization ID. To determine how the authorization ID is held, for each array element identified by the authority index, use the following defines from `sql.h`:

**SQL\_AUTH\_ORIGIN\_USER**

If this bit is on, then the authorization ID has the authority granted to the user or to a role granted to the user.

**SQL\_AUTH\_ORIGIN\_GROUP**

If this bit is on, then the authorization ID has the authority granted to the group or to a role granted to the group.

For example, to determine if a user holds DBADM authority, verify the following value:

```
authority_bitmap[SQL_DBAUTH_DBADM]
```

To determine if the DBADM authority is held directly by the user, verify the following:

```
authority_bitmap[SQL_DBAUTH_DBADM] & SQL_AUTH_ORIGIN_USER
```

## authority\_lvl - User authorization level monitor element

The highest authority level granted to an application.

**Note:** The `authority_lvl` monitor element is deprecated starting with DB2 database Version 9.5. Use the `authority_bitmap` monitor element instead. See “`authority_bitmap` - User authorization level monitor element” on page 341.

*Table 175. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic
Application	appl_info	Basic

*Table 176. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

**Usage** The operations allowed by an application are granted either directly or indirectly.

The following defines from `sql.h` may be used to determine the authorizations granted explicitly to a user:

- SQL\_SYSADM
- SQL\_DBADM
- SQL\_CREATETAB
- SQL\_BINDADD
- SQL\_CONNECT
- SQL\_CREATE\_EXT\_RT
- SQL\_CREATE\_NOT\_FENC
- SQL\_SYSCTRL
- SQL\_SYSMANT

The following defines from `sql.h` may be used to determine indirect authorizations inherited from `group` or `public`:

- `SQL_SYSADM_GRP`
- `SQL_DBADM_GRP`
- `SQL_CREATETAB_GRP`
- `SQL_BINDADD_GRP`
- `SQL_CONNECT_GRP`
- `SQL_CREATE_EXT_RT_GRP`
- `SQL_CREATE_NOT_FENC_GRP`
- `SQL_SYSCTRL_GRP`
- `SQL_SYSMANT_GRP`

---

## auto\_storage\_hybrid - Hybrid automatic storage table space indicator monitor element

If the table space is an automatic storage table space with some non-automatic storage containers, this monitor element returns a value of 1. Otherwise, it returns a value of 0.

*Table 177. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

### Usage

A hybrid automatic storage table space is a table space that has been converted to be managed by automatic storage using the `ALTER TABLESPACE` command, but has not yet been rebalanced. This table space still has non-automatic storage containers. After the table space is rebalanced, it contains only automatic storage containers, and is no longer considered a hybrid table space.

---

## automatic - Buffer pool automatic monitor element

Indicates whether a particular buffer pool has self-tuning enabled. This element is set to 1 if self-tuning is enabled for the buffer pool; and 0 otherwise.

*Table 178. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE

---

## bin\_id - Histogram bin identifier monitor element

The identifier of a histogram bin. The `bin_id` is unique within a histogram.

*Table 179. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_histogrambin	-

## Usage

Use this element to distinguish bins within the same histogram.

---

### binds\_precompiles - Binds/Precompiles Attempted

The number of binds and pre-compiles attempted.

**Element identifier**

binds\_precompiles

**Element type**

counter

*Table 180. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

*Table 181. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** You can use this element to gain insight into the current level of activity within the database manager.

This value does not include the count of *int\_auto\_rebinds*, but it does include binds that occur as a result of the REBIND PACKAGE command.

---

### block\_ios - Number of block I/O requests monitor element

The number of block I/O requests. More specifically, the number of times DB2 performs sequential prefetching of pages into the block area of the buffer pool.

*Table 182. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

*Table 183. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool	Buffer Pool

## Usage

If block-based buffer pool is enabled, this monitor element will report how often block I/O is being done. Otherwise, this monitor element will return 0. The number of block I/O requests is monitored only during sequential prefetching when using block-based buffer pools.

If block-based buffer pool is enabled and this number is very low, or close to the number of vectored I/Os (the value of the **vectored\_ios** monitor element), consider changing the block size. This state can be an indication of the following:

- The extent size of one or more table spaces bound to the buffer pool is smaller than the block size specified for the buffer pool.
- Some pages requested in the prefetch request are already present in the page area of the buffer pool.

The prefetcher allows some wasted pages in each buffer pool block, but if too many pages are wasted, then the prefetcher will decide to perform vectored I/O into the page area of the buffer pool.

To take full advantage of the sequential prefetch performance improvements that block-based buffer pools provide, it is essential to choose an appropriate value for the block size. This can, however, be difficult because multiple table spaces with different extent sizes can be bound to the same block-based buffer pool. For optimal performance, it is recommended that you bind table spaces with the same extent size to a block-based buffer pool with a block size equal to the extent size. Good performance can be achieved when the extent size of the table spaces are greater than the block size, but not when the extent size is smaller than the block size.

For example, if the extent size is 2 and the block size is 8, vectored I/O would be used instead of block I/O (block I/O would have wasted 6 pages). A reduction of the block size to 2 would solve this problem.

---

## blocking\_cursor - Blocking Cursor

This element indicates if the statement being executed is using a blocking cursor.

### Element identifier

blocking\_cursor

### Element type

information

*Table 184. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement
DCS Statement	dcs_stmt	Statement

*Table 185. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks with Details	event_detailed_dlconn	-
Statements	event_stmt	-



**Usage** Using blocking for data transfer for a query can improve its performance. The SQL used for a query can affect the use of blocking and might require some modification.

---

## blocks\_pending\_cleanup - Pending cleanup rolled-out blocks monitor element

The total number of MDC table blocks in the database that are pending asynchronous cleanup following a roll out delete.

*Table 186. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	-
Database	event_db	-

### Usage

Use this element to determine the number of MDC table blocks that, following the deletion of a defer cleanup roll out, have not been released back to the system as available storage.

---

## bottom - Histogram bin bottom monitor element

The exclusive bottom end of the range of a histogram bin. The value of this monitor element is also the top inclusive end of the range of the previous histogram bin, if there is one.

*Table 187. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_histogrambin	-

### Usage

Use this element with the corresponding **top** element to determine the range of a bin within a histogram.

---

## boundary\_leaf\_node\_splits - Boundary leaf node splits monitor element

Number of times a boundary leaf node was split during an insert operation.

*Table 188. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

---

## bp\_cur\_buffsz - Current Size of Buffer Pool

The current buffer pool size.

Table 189. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool_nodeinfo	Buffer Pool

## bp\_id - Buffer pool identifier monitor element

This element contains the buffer pool identifier for the buffer pool that is being monitored.

Table 190. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool	Basic

## bp\_name - Buffer pool name monitor element

The name of the buffer pool.

Table 191. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE

Table 192. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool	Basic

**Usage** Each database requires at least one buffer pool. Depending on your needs, you may choose to create several buffer pools, each of a different size, for a single database. The CREATE, ALTER, and DROP BUFFERPOOL statements allow you to create, change, or remove a buffer pool.

When a database is created, it has a default buffer pool called IBMDEFAULTBP with a size determined by the platform. It also has a set of system buffer pools, each corresponding to a different page size:

- IBMSYSTEMBP4K
- IBMSYSTEMBP8K
- IBMSYSTEMBP16K
- IBMSYSTEMBP32K

These system buffer pools cannot be altered.

## bp\_new\_buffsz - New Buffer Pool Size

The size the buffer pool will be changed to once the database is restarted. When the ALTER BUFFERPOOL statement is executed as DEFERRED, the buffer pool size is not changed until the database is stopped and restarted.

Table 193. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool_nodeinfo	Buffer Pool

---

## bp\_pages\_left\_to\_remove - Number of Pages Left to Remove

The number of pages left to remove from the buffer pool before the buffer pool resize is completed. This applies only to buffer pool resize operations invoked by ALTER BUFFERPOOL statements executed as IMMEDIATE.

Table 194. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool_nodeinfo	Buffer Pool

---

## bp\_tbsp\_use\_count - Number of Table Spaces Mapped to Buffer Pool

The number of table spaces using this buffer pool.

Table 195. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool_nodeinfo	Buffer Pool

---

## buff\_free - FCM Buffers Currently Free

This element indicates the number of FCM buffers currently free.

**Element identifier**

buff\_free

**Element type**

gauge

Table 196. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	fcm	Basic

**Usage** Use the number of FCM buffers currently free in conjunction with the *fcm\_num\_buffers* configuration parameter to determine the current FCM buffer pool utilization. You can use this information to tune *fcm\_num\_buffers*.

---

## buff\_free\_bottom - Minimum FCM Buffers Free

The lowest number of free FCM buffers reached during processing.

Table 197. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	fcm	Basic

**Usage** Use this element in conjunction with the *fcm\_num\_buffers* configuration parameter to determine the maximum FCM buffer pool utilization. If *buff\_free\_bottom* is low, you should increase *fcm\_num\_buffers* to ensure that operations do not run out of FCM buffers. If *buff\_free\_bottom* is high, you can decrease *fcm\_num\_buffers* to conserve system resources.

---

## byte\_order - Byte Order of Event Data

The byte ordering of numeric data, specifically whether the event data stream was generated on a “big endian” server (for example, a RS/6000®) or “little endian” server (for example, an Intel-based PC running Windows 2000).

*Table 198. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Event Log Header	event_log_header	-

**Usage** This information is needed to allow you to interpret numeric data in the data stream, since the byte order of integers on a “big endian” server is the reverse of the byte order on a “little endian” server.

If the application that processes the data recognizes that it is running on one type of computer hardware (for example, a big endian computer), while the event data was produced on the other type of computer hardware (for example, a little endian computer), then the monitoring application will have to reverse the bytes of numeric data fields before interpreting them. Otherwise, byte reordering is not required.

This element can be set to one of the following API constants:

- SQLM\_BIG\_ENDIAN
- SQLM\_LITTLE\_ENDIAN

---

## cat\_cache\_inserts - Catalog Cache Inserts

The number of times that the system tried to insert table descriptor or authorization information into the catalog cache.

**Element identifier**

cat\_cache\_inserts

**Element type**

counter

*Table 199. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

*Table 200. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** In conjunction with “Catalog Cache Lookups”, you can calculate the catalog cache hit ratio using the following formula:

$$1 - (\text{Catalog Cache Inserts} / \text{Catalog Cache Lookups})$$

See cat\_cache\_lookups for more information on using this element.

---

## cat\_cache\_lookups - Catalog Cache Lookups

The number of times that the catalog cache was referenced to obtain table descriptor information or authorization information.

### Element identifier

cat\_cache\_lookups

### Element type

counter

Table 201. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 202. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** This element includes both successful and unsuccessful accesses to the catalog cache. The catalog cache is referenced whenever:

- a table, view, or alias name is processed during the compilation of an SQL statement
- database authorization information is accessed
- a routine is processed during the compilation of an SQL statement

To calculate the catalog cache hit ratio use the following formula:

$$(1 - (\text{cat\_cache\_inserts} / \text{cat\_cache\_lookups}))$$

indicates how well the catalog cache is avoiding catalog accesses. If the ratio is high (more than 0.8), then the cache is performing well. A smaller ratio might suggest that the *catalogcache\_sz* should be increased. You should expect a large ratio immediately following the first connection to the database.

The execution of Data Definition Language (DDL) SQL statements involving a table, view, or alias will evict the table descriptor information for that object from the catalog cache causing it to be re-inserted on the next reference. In addition, GRANT and REVOKE statements for database authorization and execute privilege of routines will evict the subject authorization information from the catalog cache. Therefore, the heavy use of DDL statements and GRANT/REVOKE statements may also increase the ratio.

See the *Administration Guide* for more information on the Catalog Cache Size configuration parameter.

---

## cat\_cache\_overflows - Catalog Cache Overflows

The number of times that the catalog cache overflowed the bounds of its allocated memory.

*Table 203. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

*Table 204. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

### Usage

Use this element with the `cat_cache_size_top` monitor element to determine whether the size of the catalog cache needs to be increased to avoid overflowing.

Catalog cache space is reclaimed by evicting table descriptor information for tables, views, or aliases, or authorization information that is not currently in use by any transaction.

If the value of the `cat_cache_overflows` monitor element is large, the catalog cache may be too small for the workload. Enlarging the catalog cache may improve its performance. If the workload includes transactions which compile a large number of SQL statements referencing many tables, views, aliases, user-defined functions, or stored procedures in a single unit of work, then compiling fewer SQL statements in a single transaction may improve the performance of the catalog cache. Or if the workload includes binding of packages containing many SQL statements referencing many tables, views, aliases, user-defined functions, or stored procedures, you can try splitting packages so that they include fewer SQL statements to improve performance.

---

## cat\_cache\_size\_top - Catalog cache high watermark monitor element

The largest logical size reached by the catalog cache.

*Table 205. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

*Table 206. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

## Usage

This element indicates the maximum number of bytes the catalog cache required logically for the workload run against the database since it was activated.

The catalog cache is managed by logical size, which does not include memory management overhead. The **pool\_watermark** element in the database snapshot provides the physical high water mark value for memory used by the catalog cache. The logical size rather than physical size should be used for catalog cache monitoring and tuning efforts.

If the catalog cache overflowed, then this element contains the largest size reached by the catalog cache during the overflow. Check the **cat\_cache\_overflows** monitor element to determine if such a condition occurred.

You can determine the minimum size of the catalog cache required by your workload by:

$$\text{maximum catalog cache size} / 4096$$

Rounding the result up to a whole number, indicates the minimum number of 4K pages required by the catalog cache to avoid overflow.

---

## catalog\_node - Catalog Node Number

The node number of the node where the database catalog tables are stored.

**Element identifier**

catalog\_node

**Element type**

information

*Table 207. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

*Table 208. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

**Usage** The catalog node is the node where all system catalog tables are stored. All access to system catalog tables must go through this node.

---

## catalog\_node\_name - Catalog Node Network Name

The network name of the catalog node.

**Element identifier**

catalog\_node\_name

**Element type**

information

Table 209. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Table 210. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

**Usage** Use this element to determine the location of a database.

## ch\_free - Channels Currently Free

This element indicates the number of inter-node communication channels that are currently free.

Table 211. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	fcv	Basic

**Usage** Use the number of communication channels currently free in conjunction with the *fcv\_num\_channels* configuration parameter to determine the current connection entry utilization. You can use this information to tune *fcv\_num\_channels*.

## ch\_free\_bottom - Minimum Channels Free

The lowest number of free inter-node communication channels reached during processing.

Table 212. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	fcv	Basic

**Usage** Use this element in conjunction with the *fcv\_num\_channels* configuration parameter to determine the maximum connection entry utilization.

## client\_acctng - Client accounting string monitor element

The data passed to the target database for logging and diagnostic purposes, if the *sqleseti* API was issued in this connection. The current value of the CLIENT\_ACCTNG special register for this connection, unit of work, or activity.

This monitor element is synonymous to the **tpmon\_acc\_str** monitor element. The **client\_acctng** monitor element is used for monitoring table functions and event monitors that write to unformatted tables, which were introduced in DB2 Version 9.7. The **tpmon\_acc\_str** monitor element is used for snapshot monitors and event monitors that write to tables, files, and pipes.

Table 213. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	Always collected



Table 213. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 214. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-

## Usage

Use this element for problem determination and accounting purposes.

---

## client\_applname - Client application name monitor element

Identifies the server transaction program performing the transaction, if the sqleset API was issued in this connection. The current value of the CLIENT\_APPLNAME special register for this connection, unit of work, or activity.

This monitor element is synonymous to the **tpmon\_client\_app** monitor element. The **client\_applname** monitor element is used for monitoring table functions and event monitors that write to unformatted tables, which were introduced in DB2 Version 9.7. The **tpmon\_client\_app** monitor element is used for snapshot monitors and event monitors that write to tables, files, and pipes.

Table 215. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected
MON_GET_CONNECTION table function - Get connection metrics	Always collected

Table 215. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected

Table 216. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-

## Usage

Use this element for problem determination and accounting purposes.

---

## client\_db\_alias - Database Alias Used by Application

The alias of the database provided by the application to connect to the database.

### Element identifier

client\_db\_alias

### Element type

information

Table 217. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_id_info	Basic
Lock	appl_lock_list	Basic

Table 218. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connections	event_connheader	-

**Usage** This element can be used to identify the actual database that the application is accessing. The mapping between this name and *db\_name* could be done by using the database directories at the client node and the database manager server node.

This is the alias defined within the database manager where the database connection request originated.

This element can also be used to help you determine the authentication type, since different database aliases can have different authentication types.

---

## client\_idle\_wait\_time - Client idle wait time monitor element

This monitor element records time spent waiting for the client to send its next request. The value is given in milliseconds.

Table 219. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE

Table 220. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Use this monitor element to determine the amount of time spent working on requests, as opposed to waiting for requests from a client. A high client idle time may indicate performance issues that need to be addressed on the client rather than the server.

---

## client\_pid - Client Process ID

The process ID of the client application that made the connection to the database.

Table 221. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
Application	appl	Basic
DCS Application	dcs_appl_info	Basic

Table 222. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-
Connections	event_connheader	-

### Usage

You can use this element to correlate monitor information such as CPU and I/O time to your client application.

In the case of a DRDA AS connection, this element will be set to 0.

## client\_platform - Client Operating Platform

The operating system on which the client application is running.

Table 223. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
Application	appl	Basic
DCS Application	dcs_appl_info	Basic

Table 224. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-
Connections	event_connheader	-

### Usage

This element can be used for problem determination for remote applications. Values for this field can be found in the header file sqlmon.h.

## client\_prdid - Client product and version ID monitor element

The product and version that is running on the client.

Table 225. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_id_info	Basic
DCS Application	dcs_appl_info	Basic

Table 226. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-
Connections	event_connheader	-

## Usage

You can use this element to identify the product and code version of the IBM® data server client. It is in the form PPPVRRM, where:

- PPP identifies the product, which is “SQL” for the DB2 products
- VV identifies a 2-digit version number (with high-order 0 in the case of a 1-digit version)
- RR identifies a 2-digit release number (with high-order 0 in the case of a 1-digit release)
- M identifies a 1-character modification level (0-9 or A-Z).

---

## client\_protocol - Client Communication Protocol

The communication protocol that the client application is using to communicate with the server.

*Table 227. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
Application	appl	Basic
DCS Application	dcs_appl_info	Basic

*Table 228. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-
Connections	event_connheader	-

## Usage

This element can be used for problem determination for remote applications. Values for this field are:

### SQLM\_PROT\_UNKNOWN

The client is communicating using an unknown protocol. This value will only be returned if future clients connect with an earlier level of the server.

### SQLM\_PROT\_LOCAL

The client is running on the same node as the server and no communications protocol is in use.

### SQLM\_PROT\_TCPIP

TCP/IP

---

## client\_userid - Client user ID monitor element

The client user ID generated by a transaction manager and provided to the server, if the sqleset API is used. The current value of the CLIENT\_USERID special register for this connection, unit of work, or activity.

This monitor element is synonymous to the **tpmon\_client\_userid** monitor element. The **client\_userid** monitor element is used for monitoring table functions and event monitors that write to unformatted tables, which were introduced in DB2

Version 9.7. The **tpmon\_client\_userid** monitor element is used for snapshot monitors and event monitors that write to tables, files, and pipes.

*Table 229. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	Always collected
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

*Table 230. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-

## Usage

Use this element in application server or Transaction Processing monitor environments to identify the end-user for whom the transaction is being executed.

## client\_wrkstname - Client workstation name monitor element

Identifies the client's system or workstation (for example CICS EITERMID), if the sqleseti API was issued in this connection. The current value of the CLIENT\_WRKSTNNAME special register for this connection, unit of work, or activity.

This monitor element is synonymous to the **tpmon\_client\_wkstn** monitor element. The **client\_wrkstname** monitor element is used for monitoring table functions and event monitors that write to unformatted tables, which were introduced in DB2 Version 9.7. The **tpmon\_client\_wkstn** monitor element is used for snapshot monitors and event monitors that write to tables, files, and pipes.

*Table 231. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	Always collected
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected

Table 231. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 232. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-

## Usage

Use this element to identify the user's machine by node ID, terminal ID, or similar identifiers.

---

## codepage\_id - ID of Code Page Used by Application

The code page identifier.

Table 233. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_id_info	Basic
Lock	appl_lock_list	Basic
DCS Application	dcs_appl_info	Basic

Table 234. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Event Log Header	event_log_header	-
Connections	event_connheader	-

**Usage** For snapshot monitor data, this is the code page at the partition where the monitored application started. This identifier may be used for problem determination for remote applications. You may use this information to ensure that data conversion is supported between the application code page and the database code page (or for DRDA host databases, the host CCSID). For information about supported code pages, see the *Administration Guide*.

For event monitor data, this is the code page of the database for which event data is collected. You can use this element to determine whether your event monitor application is running under a different code page from that used by the database. Data written by the event monitor uses the database code page. If your event monitor application uses a different code page, you may need to perform some character conversion to make the data readable.

---

## comm\_private\_mem - Committed Private Memory

The amount of private memory that the instance of the database manager has currently committed at the time of the snapshot. The comm\_private\_mem value returned is only relevant on Windows operating systems.

Table 235. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

---

## commit\_sql\_stmts - Commit Statements Attempted

The total number of SQL COMMIT statements that have been attempted.

Table 236. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Database	dbase_remote	Basic
Application	appl	Basic
Application	appl_remote	Basic
DCS Database	dcs_dbase	Basic
DCS Application	dcs_appl	Basic

For snapshot monitoring, this counter can be reset.

Table 237. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** A small rate of change in this counter during the monitor period may indicate that applications are not doing frequent commits, which may lead to problems with logging and data concurrency.

You can also use this element to calculate the total number of units of work by calculating the sum of the following:

```
commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks
```

**Note:** The units of work calculated will only include those since the later of:

- The connection to the database (for database-level information, this is the time of the first connection)
- The last reset of the database monitor counters.

This calculation can be done at a database or application level.



---

## comp\_env\_desc - Compilation environment monitor element

This element stores information about the compilation environment used when compiling the SQL statement.

*Table 238. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

*Table 239. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks with Details History Values	event_stmt_history	-
Deadlocks with Details History	event_stmt_history	-
Activities	event_activitystmt	-

**Usage** This monitor element stores the compilation environment description in a binary large object. To view this information in readable form, use the COMPILATION\_ENV table function.

You can provide this element as input to the COMPILATION\_ENV table function, or to the SET COMPILATION ENVIRONMENT SQL statement.

---

## completion\_status - Completion status monitor element

The status of the unit of work.

*Table 240. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-

### Usage

Use this element to determine if the unit of work ended due to a deadlock or abnormal termination. The possible values are listed in the sqllib/misc/DB2EvmonUOW.xsd file:

- UNKNOWN
- COMMIT
- ROLLBACK
- GLOBAL\_COMMIT
- GLOBAL\_ROLLBACK
- XA\_END
- XA\_PREPARE

---

## con\_elapsed\_time - Most Recent Connection Elapsed Time

The elapsed time that the DCS application that most recently disconnected from this host database was connected.

**Element identifier**  
con\_elapsed\_time

**Element type**  
time

Table 241. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Timestamp

**Usage** Use this element as an indicator of the length of time that applications are maintaining connections to a host database.

---

## con\_local\_dbases - Local Databases with Current Connects

The number of local databases that have applications connected.

Table 242. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

**Usage** This value gives an indication of how many database information records you can expect when gathering data at the database level.

The applications can be running locally or remotely, and may or may not be executing a unit of work within the database manager

---

## con\_response\_time - Most Recent Response Time for Connect

The elapsed time between the start of connection processing and actual establishment of a connection, for the most recent DCS application that connected to this database.

**Element identifier**  
con\_response\_time

**Element type**  
time

Table 243. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Timestamp

**Usage** Use this element as an indicator of the time it currently takes applications to connect to a particular host database.

---

## concurrent\_act\_top - Concurrent activity top monitor element

The high watermark for the concurrent activities (at any nesting level) in a service subclass since the last reset.

Table 244. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-

## Usage

Use this element to know the highest concurrency of activities (including nested activities) reached on a partition for a service subclass in the time interval collected.

---

## concurrent\_connection\_top - Concurrent connection top monitor element

High watermark for concurrent coordinator connections in this service class since the last reset. This field has the same value in every subclass of the same superclass.

Table 245. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-

## Usage

This element may be useful in determining where to place thresholds on connection concurrency by showing where the current high watermark is. It is also useful for verifying that such a threshold is configured correctly and doing its job.

---

## concurrent\_wlo\_act\_top - Concurrent WLO activity top monitor element

High watermark for concurrent activities (at any nesting level) of any occurrence of this workload since the last reset.

Table 246. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wlstats	-

## Usage

Use this element to know the highest number of concurrent activities reached on a partition for any occurrence of this workload in the time interval collected.

---

## concurrent\_wlo\_top - Concurrent workload occurrences top monitor element

The high watermark for the concurrent occurrences of a workload since the last reset.

Table 247. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wlstats	-
Statistics	event_scstats	-

## Usage

Use this element to know the highest concurrency of workload occurrences reached on a partition for a workload in the time interval collected.

---

### **concurrentdbcoordactivities\_db\_threshold\_id - Concurrent database coordinator activities database threshold ID monitor element**

The ID of the CONCURRENTDBCOORDACTIVITIES database threshold that was applied to the activity.

*Table 248. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

Use this element to understand which CONCURRENTDBCOORDACTIVITIES database threshold, if any, was applied to the activity.

---

### **concurrentdbcoordactivities\_db\_threshold\_queued - Concurrent database coordinator activities database threshold queued monitor element**

This monitor element returns 'Yes' to indicate that the activity was queued by the CONCURRENTDBCOORDACTIVITIES database threshold. 'No' indicates that the activity was not queued.

*Table 249. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

Use this element to understand if the activity was queued by the CONCURRENTDBCOORDACTIVITIES database threshold applied to the activity.

---

### **concurrentdbcoordactivities\_db\_threshold\_value - Concurrent database coordinator activities database threshold value monitor element**

This monitor element returns the upper bound of the CONCURRENTDBCOORDACTIVITIES database threshold that was applied to the activity.

Table 250. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

Use this element to understand the value of the CONCURRENTDBCOORDACTIVITIES database threshold applied to the activity, if any.

---

## concurrentdbcoordactivities\_db\_threshold\_violated - Concurrent database coordinator activities database threshold violated monitor element

This monitor element returns 'Yes' to indicate that the activity violated the CONCURRENTDBCOORDACTIVITIES database threshold. 'No' indicates that the activity has not yet violated the threshold.

Table 251. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

Use this element to determine if the activity violated the CONCURRENTDBCOORDACTIVITIES database threshold that was applied to the activity.

---

## concurrentdbcoordactivities\_subclass\_threshold\_id - Concurrent database coordinator activities service subclass threshold ID monitor element

This monitor element returns the ID of the CONCURRENTDBCOORDACTIVITIES service subclass threshold that was applied to the activity.

Table 252. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

Use this element to understand which CONCURRENTDBCOORDACTIVITIES service subclass threshold, if any, was applied to the activity.

---

## **concurrentdbcoordactivities\_subclass\_threshold\_queued - Concurrent database coordinator activities service subclass threshold queued monitor element**

This monitor element returns 'Yes' to indicate that the activity was queued by the CONCURRENTDBCOORDACTIVITIES service subclass threshold. 'No' indicates that the activity was not queued.

*Table 253. Table Function Monitoring Information*

<b>Table Function</b>	<b>Monitor Element Collection Level</b>
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### **Usage**

Use this element to understand if the activity was queued by the CONCURRENTDBCOORDACTIVITIES service subclass threshold applied to the activity.

---

## **concurrentdbcoordactivities\_subclass\_threshold\_value - Concurrent database coordinator activities service subclass threshold value monitor element**

This monitor element returns the upper bound of the CONCURRENTDBCOORDACTIVITIES service subclass threshold that was applied to the activity.

*Table 254. Table Function Monitoring Information*

<b>Table Function</b>	<b>Monitor Element Collection Level</b>
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### **Usage**

Use this element to understand the value of the CONCURRENTDBCOORDACTIVITIES service subclass threshold applied to the activity, if any.

---

## **concurrentdbcoordactivities\_subclass\_threshold\_violated - Concurrent database coordinator activities service subclass threshold violated monitor element**

This monitor element returns 'Yes' to indicate that the activity violated the CONCURRENTDBCOORDACTIVITIES service subclass threshold. 'No' indicates that the activity has not yet violated the threshold.

Table 255. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

Use this element to determine if the activity violated the CONCURRENTDBCOORDACTIVITIES service subclass threshold that was applied to the activity.

---

## **concurrentdbcoordactivities\_superclass\_threshold\_id - Concurrent database coordinator activities service superclass threshold ID monitor element**

The ID of the CONCURRENTDBCOORDACTIVITIES\_SUPERCLASS threshold that was applied to the activity.

Table 256. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

Use this element to understand which CONCURRENTDBCOORDACTIVITIES service superclass threshold, if any, was applied to the activity.

---

## **concurrentdbcoordactivities\_superclass\_threshold\_queued - Concurrent database coordinator activities service superclass threshold queued monitor element**

This monitor element returns 'Yes' to indicate that the activity was queued by the CONCURRENTDBCOORDACTIVITIES service superclass threshold. 'No' indicates that the activity was not queued.

Table 257. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

Use this element to understand if the activity was queued by the CONCURRENTDBCOORDACTIVITIES service superclass threshold applied to the activity.

---

## **concurrentdbcoordactivities\_superclass\_threshold\_value - Concurrent database coordinator activities service superclass threshold value monitor element**

The upper bound of the CONCURRENTDBCOORDACTIVITIES service superclass threshold that was applied to the activity.

*Table 258. Table Function Monitoring Information*

<b>Table Function</b>	<b>Monitor Element Collection Level</b>
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### **Usage**

Use this element to understand the value of the CONCURRENTDBCOORDACTIVITIES service superclass threshold applied to the activity, if any.

---

## **concurrentdbcoordactivities\_superclass\_threshold\_violated - Concurrent database coordinator activities service superclass threshold violated monitor element**

This monitor element returns 'Yes' to indicate that the activity violated the CONCURRENTDBCOORDACTIVITIES service superclass threshold. 'No' indicates that the activity has not yet violated the threshold.

*Table 259. Table Function Monitoring Information*

<b>Table Function</b>	<b>Monitor Element Collection Level</b>
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### **Usage**

Use this element to determine if the activity violated the CONCURRENTDBCOORDACTIVITIES service superclass threshold that was applied to the activity.

---

## **concurrentdbcoordactivities\_work\_action\_set\_threshold\_id - Concurrent database coordinator activities work action set threshold ID monitor element**

The ID of the CONCURRENTDBCOORDACTIVITIES work action set threshold that was applied to the activity.

*Table 260. Table Function Monitoring Information*

<b>Table Function</b>	<b>Monitor Element Collection Level</b>
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected



## Usage

Use this element to understand which CONCURRENTDBCOORDACTIVITIES work action set threshold, if any, was applied to the activity.

---

### **concurrentdbcoordactivities\_work\_action\_set\_threshold\_queued - Concurrent database coordinator activities work action set threshold queued monitor element**

This monitor element returns 'Yes' to indicate that the activity was queued by the CONCURRENTDBCOORDACTIVITIES\_WORK\_ACTION\_SET threshold. 'No' indicates that the activity was not queued.

*Table 261. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

Use this element to understand if the activity was queued by the CONCURRENTDBCOORDACTIVITIES\_WORK\_ACTION\_SET threshold applied to the activity.

---

### **concurrentdbcoordactivities\_work\_action\_set\_threshold\_value - Concurrent database coordinator activities work action set threshold value monitor element**

The upper bound of the CONCURRENTDBCOORDACTIVITIES\_WORK\_ACTION\_SET threshold that was applied to the activity.

*Table 262. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

Use this element to understand the value of the CONCURRENTDBCOORDACTIVITIES\_WORK threshold applied to the activity, if any.

---

## concurrentdbcoordactivities\_work\_action\_set\_threshold\_violated - Concurrent database coordinator activities work action set threshold violated monitor element

This monitor element returns 'Yes' to indicate that the activity violated the CONCURRENTDBCOORDACTIVITIES\_WORK\_ACTION\_SET threshold. 'No' indicates that the activity has not yet violated the threshold.

Table 263. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### Usage

Use this element to determine if the activity violated the CONCURRENTDBCOORDACTIVITIES\_WORK\_ACTION\_SET threshold that was applied to the activity.

---

## conn\_complete\_time - Connection Request Completion Timestamp

The date and time that a connection request was granted.

### Element identifier

conn\_complete\_time

### Element type

timestamp

Table 264. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Timestamp

**Usage** Use this element to determine when a connection request to the database was granted.

---

## conn\_time - Time of database connection monitor element

The date and time of the connection to the database (at the database level, this is the first connection to the database), or when the activate database was issued.

Table 265. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-
Database	event_dbheader	-
Connections	event_connheader	-

### Usage

Use this element with the **disconn\_time** monitor element to calculate the elapsed time since:

- The database was active (for information at the database level).
- The connection was active (for information at the connection level).

---

## connection\_status - Connection Status

This element indicates the status of the communication connection status between the node issuing the GET SNAPSHOT command and other nodes listed in the *db2nodes.cfg* file.

**Element identifier**

connection\_status

**Element type**

information

*Table 266. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	fcm_node	Basic

**Usage** The connection values are :

**SQLM\_FCM\_CONNECT\_INACTIVE**

No current connection

**SQLM\_FCM\_CONNECT\_ACTIVE**

Connection is active

**SQLM\_FCM\_CONNECT\_CONGESTED**

Connection is congested

Two nodes can be active, but the communication connection between them will remain inactive, unless there is some communication between those nodes.

---

## connections\_top - Maximum Number of Concurrent Connections

The highest number of simultaneous connections to the database since the database was activated.

**Element identifier**

connections\_top

**Element type**

watermark

*Table 267. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

*Table 268. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

**Usage** You may use this element to evaluate the setting of the *maxappls* configuration parameter.

If the value of this element is the same as the *maxappls* parameter, it is likely that some database connection requests were rejected, since *maxappls* limits the number of database connections allowed.

The current number of connections at the time the snapshot was taken can be calculated using the following formula:

$$\text{rem\_cons\_in} + \text{local\_cons}$$


---

## consistency\_token - Package consistency token monitor element

For a given package name and creator, there can exist (starting in DB2 Version 8) multiple versions. The package consistency token helps to identify the version of the package that contains the SQL currently executing.

Table 269. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement

Table 270. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Statements	event_stmt	-

### Usage

You can use this element to help identify the package and the SQL statement that is executing.

---

## container\_accessible - Accessibility of container monitor element

This element indicates whether a container is accessible. A value of 1 means "Yes"; a value of 0 means "No".

Table 271. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE

Table 272. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_container	Basic

**Usage** This element can be used in conjunction with the elements **container\_id**, **container\_name**, **container\_type**, **container\_total\_pages**, **container\_usable\_pages**, and **container\_stripe\_set** to describe the container.

---

## container\_id - Container identification monitor element

An integer that uniquely defines a container within a table space.

Table 273. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONTAINER table function - Get container metrics	DATA OBJECT METRICS BASE

Table 274. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_container	Basic

**Usage** This element can be used in conjunction with the elements `container_name`, `container_type`, `container_total_pages`, `container_usable_pages`, `container_stripe_set`, and `container_accessible` to describe the container.

---

## container\_name - Container name monitor element

The name of a container.

Table 275. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONTAINER table function - Get container metrics	DATA OBJECT METRICS BASE

Table 276. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_container	Basic

**Usage** This element can be used in conjunction with the elements `container_id`, `container_type`, `container_total_pages`, `container_usable_pages`, `container_stripe_set`, and `container_accessible` to describe the container.

---

## container\_stripe\_set - Container stripe set monitor element

The stripe set that a container belongs to.

Table 277. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE

Table 278. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_container	Basic

### Usage

Use this monitor element in conjunction with the elements `container_id`, `container_name`, `container_type`, `container_total_pages`, `container_usable_pages`, and `container_accessible` to describe the container. This is only applicable to a DMS table space.

---

## container\_total\_pages - Total pages in container monitor element

The total number of pages occupied by the container.

Table 279. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE

Table 280. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_container	Basic (DMS table spaces)  Buffer Pool (SMS table spaces)

**Usage** This element can be used in conjunction with the elements `container_id`, `container_name`, `container_type`, `container_usable_pages`, `container_stripe_set`, and `container_accessible` to describe the container.

---

## container\_type - Container type monitor element

The type of the container.

Table 281. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONTAINER table function - Get container metrics	DATA OBJECT METRICS BASE

Table 282. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_container	Basic

### Usage

This element returns the type of the container, which can be a directory path (for SMS only), file (for DMS) or a raw device (for DMS). This element can be used in conjunction with the elements `container_id`, `container_name`, `container_total_pages`, `container_usable_pages`, `container_stripe_set`, and `container_accessible` to describe the container.

The valid values for this monitor element are defined in the `sqlutil.h` file.

---

## container\_usable\_pages - Usable pages in container monitor element

The total number of usable pages in a container.

Table 283. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE

Table 284. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_container	Basic (DMS table spaces) Buffer Pool (SMS table spaces)

**Usage** This element can be used in conjunction with the elements `container_id`, `container_name`, `container_type`, `container_total_pages`, `container_stripe_set`, and `container_accessible` to describe the container. For SMS table spaces, this value is the same as `container_total_pages`.

## coord\_act\_aborted\_total - Coordinator activities aborted total monitor element

The total number of coordinator activities at any nesting level that completed with errors since the last reset. For service classes, the value is updated when the activity completes. For workloads, the value is updated by each workload occurrence at the end of its unit of work.

For service classes, if you remap an activity to a different subclass with a REMAP ACTIVITY action before it aborts, then this activity counts only towards the total of the subclass it aborts in.

Table 285. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wlstats	-

### Usage

Use this element to understand if activities on the system are completing successfully. Activities may be aborted due to cancellation, errors or reactive thresholds.

## coord\_act\_completed\_total - Coordinator activities completed total monitor element

The total number of coordinator activities at any nesting level that completed successfully since the last reset. For service classes, the value is updated when the activity completes. For workloads, the value is updated by each workload occurrence at the end of its unit of work.

For service classes, if you remap an activity to a different subclass with a REMAP ACTIVITY action before it completes, then this activity counts only towards the total of the subclass it completes in.

Table 286. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wlstats	-
Statistics	event_scstats	-

## Usage

This element can be used to determine the throughput of activities in the system or to aid in calculating average activity lifetime across multiple partitions.

---

## coord\_act\_est\_cost\_avg - Coordinator activity estimated cost average monitor element

Arithmetic mean of the estimated costs for coordinator DML activities at nesting level 0 associated with this service subclass or work class since the last reset. If the internally tracked average has overflowed, the value -2 is returned. For service subclasses, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service subclass is set to NONE or BASE. For work classes, this monitor elements returns -1 if no COLLECT AGGREGATE ACTIVITY DATA EXTENDED work action is specified for the work class. For workloads, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the workload is set to NONE or BASE. Units are milliseconds.

For service classes, the estimated cost of an activity is counted only towards the service subclass in which the activity enters the system. When you remap activities between service subclasses with a REMAP ACTIVITY action, the coord\_act\_est\_cost\_avg mean of the service subclass you remap an activity to is unaffected.

Table 287. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wcstats	-
Statistics	event_wlstats	-

## Usage

Use this statistic to determine the arithmetic mean of the estimated costs of coordinator DML activities at nesting level 0 that are associated this service subclass, workload, or work class that completed or aborted since the last statistics reset.

This average can also be used to determine whether or not the histogram template used for the activity estimated cost histogram is appropriate. Compute the average activity estimated cost from the activity estimated cost histogram. Compare the computed average with this monitor element. If the computed average deviates from the true average reported by this monitor element, consider altering the histogram template for the activity estimated cost histogram, using a set of bin values that are more appropriate for your data.



---

## coord\_act\_exec\_time\_avg - Coordinator activities execution time average monitor element

Arithmetic mean of execution times for coordinator activities at nesting level 0 associated with this service subclass or work class since the last reset. If the internally tracked average has overflowed, the value -2 is returned. For service subclasses, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service subclass is set to NONE. For work classes, this monitor elements returns -1 if no COLLECT AGGREGATE ACTIVITY DATA work action is specified for the work class. For workloads, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the workload is set to NONE. Units are milliseconds.

For service classes, when you remap activities between service subclasses with a REMAP ACTIVITY action, the coord\_act\_exec\_time\_avg mean of service subclasses an activity is mapped to but does not complete in is unaffected.

Table 288. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wcstats	-
Statistics	event_wlstats	-

### Usage

Use this statistic to determine the arithmetic mean of execution time for coordinator activities associated with a service subclass, workload, or work class that completed or aborted.

This average can also be used to determine whether or not the histogram template used for the activity execution time histogram is appropriate. Compute the average activity execution time from the activity execution time histogram. Compare the computed average with this monitor element. If the computed average deviates from the true average reported by this monitor element, consider altering the histogram template for the activity execution time histogram, using a set of bin values that are more appropriate for your data.

---

## coord\_act\_interarrival\_time\_avg - Coordinator activity arrival time average monitor element

Arithmetic mean of the time between arrivals of coordinator activities at nesting level 0 associated with this service subclass or work class since the last reset. If the internally tracked average has overflowed, the value -2 is returned. For service subclasses, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service subclass is set to NONE or BASE. For work classes, this monitor elements returns -1 if no COLLECT AGGREGATE ACTIVITY DATA EXTENDED work action is specified for the work class. For workloads, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the workload is set to NONE or BASE. Units are milliseconds.

For service classes, the inter-arrival time mean is calculated for service subclasses through which activities enter the system. When you remap activities between

service subclasses with a REMAP ACTIVITY action, the coord\_act\_interarrival\_time\_avg of the service subclass you remap an activity to is unaffected.

Table 289. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wcstats	-
Statistics	event_wlstats	-

## Usage

Use this statistic to determine the arithmetic mean between arrivals of coordinator activities at nesting level 0 associated with this service subclass, workload, or work class.

The inter-arrival time can be used to determine arrival rate, which is the inverse of inter-arrival time. This average can also be used to determine whether or not the histogram template used for the activity inter-arrival time histogram is appropriate. Compute the average activity inter-arrival time from the activity inter-arrival time histogram. Compare the computed average with this monitor element. If the computed average deviates from the true average reported by this monitor element, consider altering the histogram template for the activity inter-arrival time histogram, using a set of bin values that are more appropriate for your data.

---

## coord\_act\_lifetime\_avg - Coordinator activity lifetime average monitor element

Arithmetic mean of lifetime for coordinator activities at nesting level 0 associated with this service subclass, workload, or work class since the last reset. If the internally tracked average has overflowed, the value -2 is returned. For service subclasses, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service subclass is set to NONE. For work classes, this monitor elements returns -1 if no COLLECT AGGREGATE ACTIVITY DATA work action is specified for the work class. For workloads, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the workload is set to NONE. Units are milliseconds.

For service classes, when you remap activities between service subclasses with a REMAP ACTIVITY action, only the the coord\_act\_lifetime\_avg mean of the final service class where the activity completes is affected.

Table 290. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wcstats	-
Statistics	event_wlstats	-

## Usage

Use this statistic to determine the arithmetic mean of the lifetime for coordinator activities associated with a service subclass, workload, or work class that completed or aborted.

This statistic can also be used to determine whether or not the histogram template used for the activity lifetime histogram is appropriate. Compute the average activity lifetime from the activity lifetime histogram. Compare the computed average with this monitor element. If the computed average deviates from the true average reported by this monitor element, consider altering the histogram template for the activity lifetime histogram, using a set of bin values that are more appropriate for your data.

---

## coord\_act\_lifetime\_top - Coordinator activity lifetime top monitor element

High watermark for coordinator activity lifetime, counted at all nesting levels. Units are milliseconds. For service classes, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service class is set to NONE. For work classes, this monitor element returns -1 if no COLLECT AGGREGATE ACTIVITY DATA work action is specified for the work class. For workloads, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the workload is set to NONE.

To effectively use this statistic with service classes when you also remap activities between service subclasses with a REMAP ACTIVITY action, you must aggregate the coord\_act\_lifetime\_top high watermark of any given service subclass with that of other subclasses affected by the same remapping threshold or thresholds. This is because an activity will complete after it has been remapped to a different service subclass by a remapping threshold, and the time the activity spends in other service subclasses before being remapped is counted only towards the service class in which it completes.

*Table 291. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wcstats	-
Statistics	event_scstats	-
Statistics	event_wlstats	-

## Usage

This element can be used to help determine whether or not thresholds on activity lifetime are being effective and can also help to determine how to configure such thresholds.

---

## coord\_member - Coordinator member monitor element

Coordinating member for the given unit of work or workload.

Table 292. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics	Always collected
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## coord\_act\_queue\_time\_avg - Coordinator activity queue time average monitor element

Arithmetic mean of queue time for coordinator activities at nesting level 0 associated with this service subclass or work class since the last reset. If the internally tracked average has overflowed, the value -2 is returned. For service subclasses, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service subclass is set to NONE. For work classes, this monitor elements returns -1 if no COLLECT AGGREGATE ACTIVITY DATA work action is specified for the work class. For workloads, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the workload is set to NONE. Units are milliseconds.

For service classes, the queue time counts only towards the service subclass in which the activity completes or is aborted. When you remap activities between service subclasses with a REMAP ACTIVITY action, the coord\_act\_queue\_time\_avg mean of service subclasses an activity is mapped to but does not complete in is unaffected.

### Element identifier

coord\_act\_queue\_time\_avg

### Element type

information

-->

Table 293. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wcstats	-
Statistics	event_wlstats	-

## Usage

Use this statistic to determine the arithmetic mean of the queue time for coordinator activities associated with a service subclass, workload, or work class that completed or aborted.

This statistic can also be used to determine whether or not the histogram template used for the activity queue time histogram is appropriate. Compute the average activity queue time from the activity queue time histogram. Compare the

computed average with this monitor element. If the computed average deviates from the true average reported by this monitor element, consider altering the histogram template for the activity queue time histogram, using a set of bin values that are more appropriate for your data.

---

## coord\_act\_rejected\_total - Coordinator activities rejected total monitor element

The total number of coordinator activities at any nesting level that were rejected instead of being allowed to execute since the last reset. This counter is updated when an activity is prevented from executing by either a predictive threshold or a prevent execution work action. For service classes, the value is updated when the activity completes. For workloads, the value is updated by each workload occurrence at the end of its unit of work.

*Table 294. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wlstats	-

### Usage

This element can be used to help determine whether or not predictive thresholds and work actions that prevent execution are being effective and whether or not they are too restrictive.

---

## coord\_agent\_pid - Coordinator agent identifier monitor element

The engine dispatchable unit (EDU) identifier of the coordinator agent for the application. Except on the Linux operating system, the EDU ID is mapped to the thread ID. On the Linux operating system, the EDU ID is a DB2 generated unique identifier.

*Table 295. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic

*Table 296. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-

### Usage

You can use this element to link database system monitor information to other sources of diagnostic information, such as system traces.

---

## coord\_agents\_top - Maximum Number of Coordinating Agents

The maximum number of coordinating agents working at one time.

Table 297. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic
Database	dbase	Basic

## Usage

If the peak number of coordinating agents represents too high a workload for this node, you can reduce this upper boundary by changing the `max_coordagents` configuration parameter.

---

## coord\_node - Coordinating Node

In a multi-node system, the node number of the node where the application connected or attached to the instance.

Table 298. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

Table 299. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

**Usage** Each connected application is served by one coordinator node.

---

## coord\_partition\_num - Coordinator partition number monitor element

The coordinator partition of the unit of work or activity. In a multi-partition system, the coordinator partition is the partition where the application connected to the database.

Table 300. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-
Activities	event_activity	-
Threshold violations	event_thresholdviolations	-

## Usage

This element allows the coordinator partition to be identified for activities or units of work that have records on partitions other than the coordinator.

---

## corr\_token - DRDA Correlation Token

The DRDA AS correlation token.

Table 301. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic

Table 301. Snapshot Monitoring Information (continued)

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

Table 302. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connections	event_connheader	-

**Usage** The DRDA correlation token is used for correlating the processing between the application server and the application requester. It is an identifier dumped into logs when errors arise, that you can use to identify the conversation that is in error. In some cases, it will be the LUWID of the conversation.

If communications are not using DRDA, this element returns the *appl\_id* (see *appl\_id*).

If you are using the database system monitor APIs, note that the API constant `SQLM_APPLID_SZ` is used to define the length of this element.

---

## cost\_estimate\_top - Cost estimate top monitor element

The high watermark for the estimated cost of DML activities at all nesting levels in a service subclass or work class. For service subclasses, this monitor element returns -1 when `COLLECT AGGREGATE ACTIVITY DATA` for the service subclass is set to `NONE`. For work classes, this monitor elements returns -1 if no `COLLECT AGGREGATE ACTIVITY DATA` work action is specified for the work class.

For service classes, the estimated cost of DML activities is counted only towards the service subclass in which the activity enters the system. When you remap activities between service subclasses with a `REMAP ACTIVITY` action, the `cost_estimate_top` of the service subclass you remap an activity to is unaffected.

Table 303. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wcstats	-
Statistics	event_wlstats	-

### Usage

Use this element to determine the highest DML activity estimated cost reached on a partition for a service class, workload, or work class in the time interval collected.

---

## count - Number of Event Monitor Overflows

The number of consecutive overflows that have occurred.

**Element identifier**

count

**Element type**

counter

Table 304. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Overflow Record	event_overflow	-

**Usage** You may use this element to get an indication of how much monitor data has been lost.

The event monitor sends one overflow record for a set of consecutive overflows.

---

## **cputime\_threshold\_id - CPU time threshold ID monitor element**

The ID of the CPU TIME threshold that was applied to the activity.

Table 305. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### **Usage**

Use this element to understand which CPU TIME threshold, if any, was applied to the activity.

---

## **cputime\_threshold\_value - CPU time threshold value monitor element**

The upper bound of the CPU TIME threshold that was applied to the activity.

Table 306. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### **Usage**

Use this element to understand the value of the CPU TIME threshold applied to the activity, if any.

---

## **cputime\_threshold\_violated - CPU time threshold violated monitor element**

This monitor element returns 'Yes' to indicate that the activity violated the CPU TIME threshold. 'No' indicates that the activity has not yet violated the threshold.

Table 307. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected



## Usage

Use this element to determine if the activity violated the CPUTIME threshold that was applied to the activity.

---

### **cputimeinsc\_threshold\_id - CPU time in service class threshold ID monitor element**

The ID of the CPUTIMEINSC threshold that was applied to the activity.

*Table 308. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

Use this element to understand which CPUTIMEINSC threshold, if any, was applied to the activity.

---

### **cputimeinsc\_threshold\_value - CPU time in service class threshold value monitor element**

The upper bound of the CPUTIMEINSC threshold that was applied to the activity.

*Table 309. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

Use this element to understand the value of the CPUTIMEINSC threshold applied to the activity, if any.

---

### **cputimeinsc\_threshold\_violated - CPU time in service class threshold violated monitor element**

This monitor element returns 'Yes' to indicate that the activity violated the CPUTIMEINSC threshold. 'No' indicates that the activity has not yet violated the threshold.

*Table 310. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

Use this element to determine if the activity violated the CPUTIMEINSC threshold that was applied to the activity.

---

## create\_nickname - Create Nicknames

This element contains a count of the total number of times the federated server has created a nickname over an object residing on this data source on behalf of any application since the start of the federated server instance or the last reset of the database monitor counters, whichever is latest.

*Table 311. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

## Usage

Use this element to determine the amount of CREATE NICKNAME activity against this data source by this federated server instance or an application. CREATE NICKNAME processing results in multiple queries running against the data source catalogs; therefore, if the value of this element is high, you should determine the cause and perhaps restrict this activity.

---

## create\_nickname\_time - Create Nickname Response Time

This element contains the aggregate amount of time, in milliseconds, that it has taken this data source to process CREATE NICKNAME statements from all applications or a single application running on this federated server instance. The response time is measured since the start of the federated server instance, or the last reset of the database monitor counter, whichever is the latest. The response time is measured as the difference between the time the federated server started retrieving information from the data source to process the CREATE NICKNAME statement, and the time it took to retrieve all the required data from the data source.

*Table 312. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Timestamp
Application	appl_remote	Timestamp

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to determine how much actual time was used to create nicknames for this data source.

---

## creator - Application Creator

The authorization ID of the user that pre-compiled the application.

Table 313. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement
DCS Statement	dcs_stmt	Statement

Table 314. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	event_dlconn	-
Statements	event_stmt	-
Activities	event_activitystmt	-

**Usage** Use this element to help identify the SQL statement that is processing, in conjunction with the CREATOR column of the package section information in the catalogs.

If the CURRENT PACKAGE PATH special register is set, the *creator* value may reflect different values over the lifetime of the SQL statement. If a snapshot or event monitor record is taken before PACKAGE PATH resolution, the *creator* value will reflect the value flowed in from the client request. If a snapshot or event monitor record is taken after PACKAGE PATH resolution, the *creator* value will reflect the creator of the resolved package. The resolved package will be the package whose *creator* value appears earliest in the CURRENT PACKAGE PATH SPECIAL REGISTER and whose package name and unique ID matches that of the client request.

---

## current\_active\_log - Current Active Log File Number

The file number of the active log file the DB2 database system is currently writing.

Table 315. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	detail_log	Basic

Table 316. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

**Usage** Use this element in conjunction with the *first\_active\_log* and *last\_active\_log* elements to determine the range of active log files. Knowing the range of active log files helps you determine the disk space required for log files.

You can also use this element to determine which log files have data to help you identify log files needed for split mirror support.

---

## current\_archive\_log - Current Archive Log File Number

The file number of the log file the DB2 database system is currently archiving. If the DB2 database system is not archiving a log file, the value for this element is SQLM\_LOGFILE\_NUM\_UNKNOWN.

Table 317. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	detail_log	Basic

Table 318. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

**Usage** Use this element to determine if there is a problem archiving log files. Such problems include:

- Slow archive media
- Archive media that is not available

---

## current\_extent - Extent currently being moved monitor element

The numeric identifier of the extent currently being moved by the table space rebalancing process.

Table 319. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_EXTENT_MOVEMENT_STATUS - Get extent movement progress status metrics	Always collected

---

## cursor\_name - Cursor Name

The name of the cursor corresponding to this SQL statement.

**Element identifier**

cursor\_name

**Element type**

information

Table 320. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement

Table 321. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks with Details	event_detailed_dlconn	-
Statements	event_stmt	-

**Usage** You may use this element to identify the SQL statement that is processing. This name will be used on an OPEN, FETCH, CLOSE, and PREPARE of an SQL SELECT statement. If a cursor is not used, this field will be blank.

---

## data\_object\_pages - Data Object Pages

The number of disk pages consumed by a table. This size represents the base table size only. Space consumed by index objects, LOB data, and long data are reported by *index\_object\_pages*, *lob\_object\_pages*, and *long\_object\_pages*, respectively.

Table 322. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table	Basic

Table 323. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Tables	event_table	-

**Usage** This element provides a mechanism for viewing the actual amount of space consumed by a particular table. This element can be used in conjunction with a table event monitor to track the rate of table growth over time.

---

## data\_partition\_id - Data partition identifier monitor element

The identifier of the data partition for which information is returned.

Table 324. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLE table function - Get table metrics	Always collected
MON_GET_INDEX table function - Get index metrics	Always collected

Table 325. Snapshot monitoring information

Snapshot level	Logical data grouping	Monitor switch
Table	table	Basic
Lock	lock	Lock
Lock	lock_wait	Lock

Table 326. Event monitoring information

Event type	Logical data grouping	Monitor switch
Table	event_table	-
Deadlocks	event_dlconn	-
Deadlocks with Details	event_detailed_dlconn	-
Deadlocks	lock	-

### Usage

This element is only applicable to partitioned tables and partitioned indexes. Otherwise, the value of this monitor element is NULL.

When returning lock level information, a value of -1 represents a lock which controls access to the whole table.

---

## datasource\_name - Data Source Name

This element contains the name of the data source whose remote access information is being displayed by the federated server. This element corresponds to the 'SERVER' column in SYSCAT.SERVERS.

**Element identifier**

datasource\_name

**Element type**

information

*Table 327. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Basic
Application	appl_remote	Basic

**Usage** Use this element to identify the data source whose access information has been collected and is being returned.

---

## db2\_status - Status of DB2 Instance

The current status of the instance of the database manager.

*Table 328. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

**Usage** You can use this element to determine the state of your database manager instance.

Values for this element are:

API Constant	Value	Description
SQLM_DB2_ACTIVE	0	The database manager instance is active.
SQLM_DB2_QUIESCE_PEND	1	The instance and the databases in the instance are in quiesce-pending state. New connections to any instance database are not permitted and new units of work cannot be started. Depending on the quiesce request, active units of work will be allowed to complete or will be rolled back immediately.
SQLM_DB2_QUIESCED	2	The instance and the databases in the instance has been quiesced. New connections to any instance database are not permitted and new units of work cannot be started.

---

## db2start\_time - Start Database Manager Timestamp

The date and time that the database manager was started using the db2start command.

**Element identifier**  
db2start\_time

**Element type**  
timestamp

Table 329. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

**Usage** This element may be used with the *time\_stamp* monitor element to calculate the elapsed time since the database manager was started up until the snapshot was taken.

---

## db\_conn\_time - Database activation timestamp monitor element

The date and time of the connection to the database (at the database level, this is the first connection to the database), or when the activate database was issued.

Table 330. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Timestamp
Table Space	tablespace_list	Buffer Pool, Timestamp
Table	table_list	Timestamp

Table 331. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-

### Usage

Use this element with the **disconn\_time** monitor element to calculate the total connection time.

---

## db\_heap\_top - Maximum Database Heap Allocated

This element is being maintained for DB2 version compatibility. It now measures memory usage, but not exclusively usage by the database heap.

**Note:** The **db\_heap\_top** monitor element is deprecated starting with DB2 Version 9.5. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

Table 332. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Table 333. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

## db\_location - Database Location

The location of the database in relation to the application.

Table 334. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

**Usage** Determine the relative location of the database server with respect to the application taking the snapshot. Values are:

- SQLM\_LOCAL
- SQLM\_REMOTE

## db\_name - Database Name

The real name of the database for which information is collected or to which the application is connected. This is the name the database was given when created.

Table 335. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Database	dbase_remote	Basic
Application	appl_id_info	Basic
Application	appl_remote	Basic
Table Space	tablespace_list	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Table	table_list	Table
Lock	db_lock_list	Basic
Dynamic SQL	dynsql_list	Basic
DCS Database	dcs_dbase	Basic
DCS Application	dcs_appl_info	Basic

Table 336. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_dbheader	-

**Usage** You may use this element to identify the specific database to which the data applies.

For applications that are not using DB2 Connect to connect to a host or System i<sup>®</sup> database server, you can use this element in conjunction with the **db\_path** monitor element to uniquely identify the database and help relate the different levels of information provided by the monitor.



---

## db\_path - Database Path

The full path of the location where the database is stored on the monitored system.

Table 337. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl_id_info	Basic
Table Space	tablespace_list	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Table	table_list	Table
Lock	db_lock_list	Basic
Dynamic SQL	dynsql_list	Basic

Table 338. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_dbheader	-

**Usage** This element can be used with the *db\_name* monitor element to identify the specific database to which the data applies.

---

## db\_status - Status of Database

The current status of the database.

Table 339. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

**Usage** You can use this element to determine the state of your database.

Values for this field are:

API Constant	Value	Description
SQLM_DB_ACTIVE	0	The database is active.
SQLM_DB QUIESCE_PEND	1	The database is in quiesce-pending state. New connections to the database are not permitted and new units of work cannot be started. Depending on the quiesce request, active units of work will be allowed to complete or will be rolled back immediately.
SQLM_DB QUIESCED	2	The database has been quiesced. New connections to the database are <b>not</b> permitted and new units of work cannot be started.
SQLM_DB_ROLLFWD	3	A rollforward is in progress on the database.

---

## db\_storage\_path - Automatic storage path monitor element

This element shows the full path of a location that is used by the database for placing automatic storage table spaces. There can be 0 or more storage paths associated with a database.

Table 340. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	db_sto_path_info	Basic

### Usage

Use this element with the **num\_db\_storage\_paths** monitor element to identify the storage paths that are associated with this database.

---

## db\_storage\_path\_state - Storage path state monitor element

The automatic storage path state indicates whether the storage path is in use by the database.

Table 341. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	db_sto_path_info	Basic

### Usage

Use this monitor element to determine whether the storage path is in use by the database. The following values are possible:

#### NOT\_IN\_USE

There are no table spaces using this storage path on the specified database partition.

#### IN\_USE

There are table spaces using this storage path on the specified database partition.

#### DROP\_PENDING

This storage path has been dropped, but some table spaces are still using it. Before storage paths are physically dropped from the database, all table spaces must stop using them. To stop using a dropped storage path, either drop the table space or rebalance the table space using the REBALANCE clause of the ALTER TABLESPACE statement.

---

## db\_storage\_path\_with\_dpe - Storage path including database partition expression monitor element

Automatic storage path that includes the unevaluated database partition expression.

Table 342. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	db_sto_path_info	Basic

## Usage

Use this monitor element to determine the storage path that was specified for the database as part of the CREATE DATABASE command or the ALTER DATABASE statement, if the storage path contains a database partition expression.

If the storage path does not contain a database partition expression, then this monitor element returns a null value.

---

## db\_work\_action\_set\_id - Database work action set ID monitor element

If this activity has been categorized into a work class of database scope, this monitor element shows the ID of the work action set associated with the work class set to which the work class belongs. Otherwise, this monitor element shows the value of 0.

Table 343. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
WLM_GET_ACTIVITY_DETAILS_COMPLETE (reported in DETAILS XML document)	Always collected

Table 344. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

## Usage

This element can be used with the **db\_work\_class\_id** element to uniquely identify the database work class of the activity, if one exists.

---

## db\_work\_class\_id - Database work class ID monitor element

If this activity has been categorized into a work class of database scope, this monitor element displays the ID of the work class. Otherwise, this monitor element displays the value of 0.

Table 345. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
WLM_GET_ACTIVITY_DETAILS_COMPLETE table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 346. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

## Usage

This element can be used with the **db\_work\_action\_set\_id** element to uniquely identify the database work class of the activity, if one exists.

---

## dcx\_appl\_status - DCS Application Status

The status of a DCS application at the DB2 Connect gateway.

Table 347. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcx_appl_info	Basic

**Usage** Use this element for problem determination on DCS applications. Values are:

- **SQLM\_DCS\_CONNECTPEND\_OUTBOUND**  
The application has initiated a database connection from the DB2 Connect gateway to the host database, but the request has not completed yet.
- **SQLM\_DCS\_UOWWAIT\_OUTBOUND**  
The DB2 Connect gateway is waiting for the host database to reply to the application's request.
- **SQLM\_DCS\_UOWWAIT\_INBOUND**  
The connection from the DB2 Connect gateway to the host database has been established and the gateway is waiting for SQL requests from the application. Or the DB2 Connect gateway is waiting on behalf of the unit of work in the application. This usually means that the application's code is being executed.

---

## dcx\_db\_name - DCS Database Name

The name of the DCS database as cataloged in the DCS directory.

Table 348. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcx_dbase	Basic
DCS Application	dcx_appl_info	Basic

**Usage** Use this element for problem determination on DCS applications.

---

## ddl\_sql\_stmts - Data Definition Language (DDL) SQL Statements

This element indicates the number of SQL Data Definition Language (DDL) statements that were executed.

Table 349. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 350. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Table 350. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

**Usage** You can use this element to determine the level of database activity at the application or database level. DDL statements are expensive to run due to their impact on the system catalog tables. As a result, if the value of this element is high, you should determine the cause, and possibly restrict this activity from being performed.

You can also use this element to determine the percentage of DDL activity using the following formula:

$$\text{ddl\_sql\_stmts} / \text{total number of statements}$$

This information can be useful for analyzing application activity and throughput. DDL statements can also impact:

- the catalog cache, by invalidating table descriptor information and authorization information that are stored there and causing additional system overhead to retrieve the information from the system catalogs
- the package cache, by invalidating sections that are stored there and causing additional system overhead due to section recompilation.

Examples of DDL statements are CREATE TABLE, CREATE VIEW, ALTER TABLE, and DROP INDEX.

## deadlock\_id - Deadlock Event Identifier

The deadlock identifier for a deadlock.

**Element identifier**

deadlock\_id

**Element type**

information

Table 351. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	event_deadlock	-
Deadlocks	event_dlconn	-
Deadlocks with Details	event_detailed_dlconn	-
Deadlocks with Details History	event_detailed_dlconn	-
Deadlocks with Details History	event_stmt_history	-
Deadlocks with Details History Values	event_data_value	-
Deadlocks with Details History Values	event_detailed_dlconn	-
Deadlocks with Details History Values	event_stmt_history	-

**Usage** Use this element in your monitoring application to correlate deadlock connection and statement history event records with deadlock event records.

---

## deadlock\_node - Partition Number Where Deadlock Occurred

Partition number where the deadlock occurred.

**Element identifier**  
deadlock\_node

**Element type**  
information

*Table 352. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	event_deadlock	-
Deadlocks	event_dlconn	-
Deadlocks with Details	event_detailed_dlconn	-

**Usage** This element is relevant only for partitioned databases. Use this in your monitoring application to correlate deadlock connection event records with deadlock event records.

---

## deadlocks - Deadlocks detected monitor element

The total number of deadlocks that have occurred.

*Table 353. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 353. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 354. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Lock

For snapshot monitoring, this counter can be reset.

Table 355. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Connection	event_conn	-

## Usage

This element can indicate that applications are experiencing contention problems. These problems could be caused by the following situations:

- Lock escalations are occurring for the database
- An application may be locking tables explicitly when system-generated row locks may be sufficient
- An application may be using an inappropriate isolation level when binding
- Catalog tables are locked for repeatable read
- Applications are getting the same locks in different orders, resulting in deadlock.

You may be able to resolve the problem by determining in which applications (or application processes) the deadlocks are occurring. You may then be able to modify the application to better enable it to run concurrently. Some applications, however, may not be capable of running concurrently.

You can use the connection timestamp monitor elements (**last\_reset**, **db\_conn\_time**, and **appl\_con\_time**) to determine the severity of the deadlocks. For example, 10 deadlocks in 5 minutes is much more severe than 10 deadlocks in 5 hours.

The descriptions for the related elements listed above may also provide additional tuning suggestions.

---

## degree\_parallelism - Degree of Parallelism

The degree of parallelism requested when the query was bound.

**Element identifier**

degree\_parallelism

**Element type**

information

*Table 356. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement

**Usage** Use with `agents_top`, to determine if the query achieved maximum level of parallelism.

---

## del\_keys\_cleaned - Pseudo deleted keys cleaned monitor element

Number of pseudo deleted keys that have been cleaned.

*Table 357. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

---

## delete\_sql\_stmts - Deletes

This element contains a count of the total number of times the federated server has issued a DELETE statement to this data source on behalf of any application since the start of the federated server instance or the last reset of the database monitor counters, whichever is latest.

*Table 358. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to determine the level of database activity directed against this data source by the federated server or an application.

You can also use this element to determine the percentage of write activity against this data source by the federated server or an application, with the following formula:

$$\text{write\_activity} = \frac{(\text{INSERT statements} + \text{UPDATE statements} + \text{DELETE statements})}{(\text{SELECT statements} + \text{INSERT statements} + \text{UPDATE statements} + \text{DELETE statements})}$$



---

## delete\_time - Delete Response Time

This element contains the aggregate amount of time, in milliseconds, that it has taken this data source to respond to DELETES from all applications or a single application running on this federated server instance since the start of the federated server instance or the last reset of the database monitor counters, whichever is latest.

The response time is measured as the difference in time between the time the federated server submits a DELETE statement to the data source, and the time the data source responds to the federated server, indicating the DELETE has been processed.

*Table 359. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Timestamp
Application	appl_remote	Timestamp

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to determine how much actual time transpires while waiting for DELETES to this data source to be processed. This information can be useful for capacity planning and tuning.

---

## destination\_service\_class\_id – Destination service class ID monitor element

The ID of the service subclass to which an activity was remapped when the threshold violation record to which this element belongs was generated. This element has a value of zero for any threshold action other than REMAP ACTIVITY.

**Element identifier**

destination\_service\_class\_id

**Element type**

Information

*Table 360. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Threshold violations	event_thresholdviolations	-

**Usage**

Use this element to trace the path of an activity through the service classes to which it was remapped. This element can also be used to compute aggregates of how many activities were mapped into a given service subclass.

---

## diaglog\_write\_wait\_time - Diagnostic log file write wait time monitor element

The time spent waiting on a write to the db2diag log file. The value is given in milliseconds.

Table 361. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

Table 362. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Use this element to understand the amount of time spent writing to the db2diag log file. In a partitioned database environment, a high value for this time may indicate contention for the db2diag log file if shared storage is being used for the diagpath. A high value may also indicate excessive logging, for example if **diaglevel** has been set to log all informational messages.

## diaglog\_writes\_total - Total diagnostic log file writes monitor element

The number of times agents have written to the db2diag log file.

Table 363. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 363. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

Table 364. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Use this element with `diaglog_write_wait_time` to understand the average amount of time spent writing to the `db2diag` log file.

## direct\_read\_reqs - Direct read requests monitor element

The number of requests to perform a direct read of one or more sectors of data.

Table 365. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 365. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 366. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 367. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Connection	event_conn	-
Tablespaces	event_tablespace	-

## Usage

Use the following formula to calculate the average number of sectors that are read by a direct read:

$$\text{direct\_reads} / \text{direct\_read\_reqs}$$

## direct\_read\_time - Direct read time monitor element

The elapsed time required to perform the direct reads. This value is given in milliseconds.

Table 368. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 369. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 370. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Connection	event_conn	-
Tablespaces	event_tablespace	-

## Usage

Use the following formula to calculate the average direct read time per sector:

$\text{direct\_read\_time} / \text{direct\_reads}$

A high average time may indicate an I/O conflict.

---

## direct\_reads - Direct reads from database monitor element

The number of read operations that do not use the buffer pool.

Table 371. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 371. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 372. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 373. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Connection	event_conn	-
Tablespaces	event_tablespace	-

## Usage

Use the following formula to calculate the average number of sectors that are read by a direct read:

$$\text{direct\_reads} / \text{direct\_read\_reqs}$$

When using system monitors to track I/O, this element helps you distinguish database I/O from non-database I/O on the device.

Direct reads are performed in units, the smallest being a 512-byte sector. They are used when:

- Reading LONG VARCHAR columns
- Reading LOB (large object) columns
- Performing a backup

---

## direct\_write\_reqs - Direct write requests monitor element

The number of requests to perform a direct write of one or more sectors of data.

*Table 374. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

*Table 375. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool



For snapshot monitoring, this counter can be reset.

*Table 376. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Connection	event_conn	-
Tablespaces	event_tablespace	-

## Usage

Use the following formula to calculate the average number of sectors that are written by a direct write:

$$\text{direct\_writes} / \text{direct\_write\_reqs}$$

## direct\_write\_time - Direct write time monitor element

The elapsed time required to perform the direct writes. This value is reported in milliseconds.

*Table 377. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 377. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 378. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 379. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Connection	event_conn	-
Tablespaces	event_tablespace	-

## Usage

Use the following formula to calculate the average direct write time per sector:

$$\text{direct\_write\_time} / \text{direct\_writes}$$

A high average time may indicate an I/O conflict.

## direct\_writes - Direct writes to database monitor element

The number of write operations that do not use the buffer pool.

*Table 380. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

*Table 381. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 382. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Connection	event_conn	-
Tablespaces	event_tablespace	-

## Usage

Use the following formula to calculate the average number of sectors that are written by a direct write.

$$\text{direct\_writes} / \text{direct\_write\_reqs}$$

When using system monitors to track I/O, this element helps you distinguish database I/O from non-database I/O on the device.

Direct writes are performed in units, the smallest being a 512-byte sector. They are used when:

- Writing LONG VARCHAR columns
- Writing LOB (large object) columns
- Performing a restore
- Performing a load
- Allocating new extents for SMS table space if MPFA is enabled (which is the default)

---

## disconn\_time - Database Deactivation Timestamp

The date and time that the application disconnected from the database (at the database level, this is the time the last application disconnected).

### Element identifier

disconn\_time

### Element type

timestamp

Table 383. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** Use this element to calculate the elapsed time since:

- The database was active (for information at the database level)

- The connection was active (for information at the connection level).

---

## disconnects - Disconnects

This element contains a count of the total number of times the federated server has disconnected from this data source on behalf of any application since the start of the federated server instance or the last reset of the database monitor counters, whichever is latest.

*Table 384. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Basic

For snapshot monitoring, this counter can be reset.

### Usage

Use this element to determine the total number of times the federated server has disconnected from this data source on behalf of any application. Together with the CONNECT count, this element provides a mechanism by which you can determine the number of applications this instance of the federated server believes is currently connected to a data source.

---

## dl\_conns - Connections involved in deadlock monitor element

The number of connections that are involved in the deadlock.

*Table 385. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks <sup>1</sup>	event_deadlock	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

### Usage

Use this element in your monitoring application to identify how many deadlock connection event records will follow in the event monitor data stream.

---

## dynamic\_sql\_stmts - Dynamic SQL Statements Attempted

The number of dynamic SQL statements that were attempted.

*Table 386. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 387. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** You can use this element to calculate the total number of successful SQL statements at the database or application level:

```

dynamic_sql_stmts
+ static_sql_stmts
- failed_sql_stmts
= throughput during monitoring period

```

## eff\_stmt\_text - Effective statement text monitor element

The effective text of the SQL statement, if the statement was modified as a result of the statement concentrator.

Table 388. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
WLM_GET_ACTIVITY_DETAILS_COMPLETE table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 389. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activity	event_activystmt	-

### Usage

If the statement concentrator is enabled and if the statement text has been modified as a result of the statement concentrator, then this monitor element contains the effective statement text. Otherwise, this monitor element contains a text string which is 0 bytes long.

## effective\_isolation - Effective isolation monitor element

The effective isolation level for this activity.

Table 390. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected

Table 391. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-

## Usage

Use this element to understand the isolation level that was used during the execution of the activity.

---

### effective\_lock\_timeout - Effective lock timeout monitor element

The effective lock timeout value for this activity.

Table 392. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

---

### effective\_query\_degree - Effective query degree monitor element

The effective query degree of parallelism for this activity.

Table 393. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 394. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-

---

### elapsed\_exec\_time - Statement Execution Elapsed Time

At the DCS statement level, this is the elapsed time spent processing an SQL request on a host database server. This value is reported by this server. In contrast to the host\_response\_time element, this element does not include the network elapsed time between DB2 Connect and the host database server. At other levels, this value represents the sum of the host execution times for all the statements that were executed for a particular database or application, or for those statements that used a given number of data transmissions.

Table 395. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Statement, Timestamp
Application	appl	Statement, Timestamp
DCS Database	dcs_dbase	Statement, Timestamp
DCS Application	dcs_appl	Statement, Timestamp
DCS Statement	dcs_stmt	Statement, Timestamp

Table 395. Snapshot Monitoring Information (continued)

Snapshot Level	Logical Data Grouping	Monitor Switch
Data Transmission	stmt_transmissions	Statement, Timestamp

For snapshot monitoring at the statement level, this counter cannot be reset. This counter can be reset at other levels.

**Usage** Use this element, along with other elapsed time monitor elements, to evaluate the database server's processing of SQL requests and to help isolate performance issues.

Subtract this element from the `host_response_time` element to calculate the network elapsed time between DB2 Connect and the host database server.

**Note:** For the `dcx_dbase`, `dcx_appl`, `dcx_stmt` and `stmt_transmissions` levels, the *elapsed\_exec\_time* element applies only to z/OS® databases. If the DB2 Connect gateway is connecting to a Windows, Linux, AIX, or other UNIX database, the *elapsed\_exec\_time* is reported as zero.

---

## empty\_pages\_deleted - Empty pages deleted monitor element

All keys on pseudo empty pages have been pseudo deleted. This monitor element reports the number of pseudo empty pages that have been deleted.

Table 396. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

---

## empty\_pages\_reused - Empty pages reused monitor element

All keys on pseudo empty pages have been pseudo deleted. This monitor element reports the number of pseudo empty pages that have been reused.

Table 397. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

---

## entry\_time - Entry time monitor element

The time at which this activity entered the system.

Table 398. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### Usage



---

## estimatedsqlcost\_threshold\_id - Estimated SQL cost threshold ID monitor element

The ID of the ESTIMATEDSQLCOST threshold that was applied to the activity.

*Table 399. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### Usage

Use this element to understand which ESTIMATEDSQLCOST threshold, if any, was applied to the activity.

---

## estimatedsqlcost\_threshold\_value - Estimated SQL cost threshold value monitor element

The upper bound of the ESTIMATEDSQLCOST threshold that was applied to the activity.

*Table 400. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### Usage

Use this element to understand the value of the ESTIMATEDSQLCOST threshold applied to the activity, if any.

---

## estimatedsqlcost\_threshold\_violated - Estimated SQL cost threshold violated monitor element

This monitor element returns 'Yes' to indicate that the activity violated the ESTIMATEDSQLCOST threshold. 'No' indicates that the activity has not yet violated the threshold.

*Table 401. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### Usage

Use this element to determine if the activity violated the ESTIMATEDSQLCOST threshold that was applied to the activity.

---

## event\_monitor\_name - Event Monitor Name

The name of the event monitor that created the event data stream.

**Element identifier**

event\_monitor\_name

**Element type**

information

*Table 402. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Event Log Header	event_log_header	-

**Usage** This element allows you to correlate the data that you are analyzing to a specific event monitor in the system catalog tables. This is the same name that can be found in the NAME column of the SYSCAT.EVENTMONITORS catalog table, which is the name specified on the CREATE EVENT MONITOR and SET EVENT MONITOR statements.

---

## event\_time - Event Time

The date and time an event occurred.

*Table 403. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Tablespaces	event_tablespace	-
Tables	event_table	-

**Usage** You can use this element to help relate events chronologically.

---

## evmon\_activates - Number of Event Monitor Activations

The number of times an event monitor has been activated.

**Element identifier**

evmon\_activates

**Element type**

counter

*Table 404. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tables	event_table	-
Tablespaces	event_tablespace	-
Bufferpools	event_bufferpool	-
Deadlocks	event_deadlock	-
Deadlocks	event_dlconn	-
Deadlocks with Details	event_detailed_dlconn	-

**Usage** Use this element to correlate information returned by the above event

types. This element is applicable only to write-to-table event monitors. This monitor element is not maintained for event monitors that write to a file or pipe.

Only some types of write-to-table event monitors use the `evmon_activates` monitor element (the event monitor types that do use this element are listed in the previous table, "Event Monitoring Information"). These event monitors update the `evmon_activates` column of the `SYSCAT.EVENTMONITORS` catalog table when activated. This change is logged, so the DATABASE CONFIGURATION will display:

```
Database is consistent = NO
```

If an event monitor is created with the `AUTOSTART` option, and the first user `CONNECTS` to the database and immediately `DISCONNECTS` so that the database is deactivated, a log file will be produced.

## executable\_id - Executable ID monitor element

An opaque binary token generated on the data server that uniquely identifies the SQL statement section that was executed. For non-SQL activities, a 0-length string value is returned.

*Table 405. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

*Table 406. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Activity	event_activitystmt	-

### Usage

Use this monitor element as input to different monitoring interfaces to obtain data about the section. The `MON_GET_PKG_CACHE_STMT` table function, which is used to get SQL statement activity metrics in the package cache, takes the executable ID as input.

## evmon\_flushes - Number of Event Monitor Flushes

The number of times the `FLUSH EVENT MONITOR SQL` statement has been issued.

### Element identifier

`evmon_flushes`

### Element type

information

Table 407. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tables	event_table	-
Tablespaces	event_tablespace	-
Bufferpools	event_bufferpool	-

**Usage** This identifier increments with each successive FLUSH EVENT MONITOR SQL request processed by the database manager after an application has connected to the database. This element helps to uniquely identify database, table, table space and buffer pool data.

---

## execution\_id - User Login ID

The ID that the user specified when logging in to the operating system. This ID is distinct from auth\_id, which the user specifies when connecting to the database.

Table 408. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
Application	appl	Basic
DCS Application	dcx_appl_info	Basic

Table 409. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connections	event_connheader	-

**Usage** You can use this element to determine the operating system userid of the individual running the application that you are monitoring.

---

## failed\_sql\_stmts - Failed Statement Operations

The number of SQL statements that were attempted, but failed.

Table 410. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Database	dbase_remote	Basic
Application	appl	Basic
Application	appl_remote	Basic
DCS Database	dcx_dbase	Basic
DCS Application	dcx_appl	Basic

For snapshot monitoring, this counter can be reset.

Table 411. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Table 411. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

**Usage** You can use this element to calculate the total number of successful SQL statements at the database or application level:

```

dynamic_sql_stmts
+ static_sql_stmts
- failed_sql_stmts
= throughput during monitoring period

```

This count includes all SQL statements that received a negative SQLCODE.

This element may also help you in determining reasons for poor performance, since failed statements mean time wasted by the database manager and as a result, lower throughput for the database.

## fcm\_message\_recv\_volume - FCM message received volume monitor element

The amount of data received for internal requests (such as RPCs) distributed by the FCM communications layer. This value is reported in bytes.

Table 412. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

Table 413. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE

Table 413. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Use this element to determine how much of the data volume being sent through the FCM subsystem is for request or reply message traffic as opposed to actual table data.

---

## fcm\_message\_rcv\_wait\_time - FCM message received wait time monitor element

The time spent by an agent waiting for an FCM reply message containing the results of a previously sent FCM request message. This time reflects both the time required to process the request message on the other end, as well as the time required to send the response between partitions using FCM. The value is given in milliseconds.

Table 414. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

Table 415. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

This element can be used to determine how much time was spent on a given partition waiting for requests to be processed on other partitions in a multi-partition instance.

---

## fcm\_message\_recvs\_total - Total FCM message receives monitor element

The total number of buffers received as part of an FCM reply message containing the results of a previously sent FCM request message.

*Table 416. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

## Usage

This element can be used to determine both the average volume per FCM message received, as well as the average time spent waiting for a single FCM message to be received.

---

## fcm\_message\_send\_volume - FCM message send volume monitor element

Amount of data volume sent via internal FCM requests. This value is reported in bytes.

*Table 417. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 417. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

## Usage

Use this element to determine how much of the data volume being sent through the FCM subsystem is used for sending request and reply message traffic, as opposed to sending actual table data.

## fcm\_message\_send\_wait\_time - FCM message send wait time monitor element

The time spent blocking on an FCM message send. The value is given in milliseconds. This monitor element reflects the time spent blocking for FCM buffers to be flushed from an FCM channel when distributing internal requests on the database system.

Table 418. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

## Usage

Use this element to determine how much time agents are spending waiting to send an FCM request message through the FCM subsystem. Depending on how busy



the FCM daemons are, an agent may need to wait when attempting to send messages.

---

## fcm\_message\_sends\_total - Total FCM message sends monitor element

The total number of buffers distributed as part of internal requests using the FCM communications mechanism.

*Table 419. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

*Table 420. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

### Usage

Use this element to determine both the average amount of data sent per FCM request message, as well as the average amount of time waited per FCM message.

---

## fcm\_rcv\_volume - FCM received volume monitor element

The total amount of data received via the FCM communications layer. This value is reported in bytes.

Table 421. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 422. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Indicates the total volume of data received on this partition using FCM, including both message traffic and table queue data.

## **fcm\_rcv\_wait\_time - FCM received wait time monitor element**

The total time spent waiting to receive data through FCM. The value is given in milliseconds.

Table 423. Table Function Monitoring Information

Table Function	Monitor Element	Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS	BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS	BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS	BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS	BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS	BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS	BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS	BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS	BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS	BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS	BASE

Table 424. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Use this element to determine the total time spent waiting to receive data through FCM on this database partition. This includes both data from replies to request messages as well as table queue data.

## fcm\_recvs\_total - FCM receives total monitor element

Total number of buffers received for internal requests using the FCM communications mechanism. The fcm\_recvs\_total monitor element value is the sum of the values for the fcm\_message\_recvs\_total and fcm\_tq\_recvs\_total monitor elements.

Table 425. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 426. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Use this element together with the `fcm_recv_wait_time` monitor element to determine the average wait time per FCM receive operation as well as the average volume returned from an FCM receive operation.

---

## fcm\_send\_volume - FCM send volume monitor element

The total amount of data distributed by the FCM communications layer. This value is reported in bytes.

*Table 427. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

## Usage

Use this monitor element to determine the total volume of data sent using FCM, including both message traffic and table queue data.

---

## fcm\_send\_wait\_time - FCM send wait time monitor element

The time spent blocking on an FCM send operation. This includes time spent waiting for buffers for internal requests to be flushed and time spent waiting for window count acknowledgements when sending data over table queues. The value is given in milliseconds.

Table 428. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 429. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Use this element to determine the total time spent waiting to send data through FCM. This includes both request messages and table queue data.

---

### **fcm\_sends\_total - FCM sends total monitor element**

The total number of buffers sent using the internal FCM communications layer.

Table 430. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 431. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Use this element to determine the average wait time per FCM receive operation as well as the average volume returned from an FCM receive operation.

---

## fcm\_tq\_recv\_volume - FCM table queue received volume monitor element

The amount of data received on table queues by the FCM communications layer. This value is reported in bytes.

*Table 432. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

*Table 433. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

### Usage

Use this monitor element to determine the total volume of data received through table queues.

---

## fcm\_tq\_recv\_wait\_time - FCM table queue received wait time monitor element

The time spent waiting to receive the next buffer from a table queue. The value is given in milliseconds.



Table 434. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

Table 435. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Use this element to determine how much time agents are spending waiting to receive data on table queues.

## fcm\_tq\_recvs\_total - FCM table queue receives total monitor element

The total number of buffers received from table queues using the internal FCM communications mechanism.

Table 436. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 436. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

Table 437. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Use this element in conjunction with **fcm\_tq\_rcv\_volume** and **fcm\_tq\_rcv\_wait\_time** to determine the average wait time and volume per table queue buffer received.

## **fcm\_tq\_send\_volume - FCM table queue send volume monitor element**

The amount of data sent over table queues by the FCM communications layer. This value is reported in bytes.

Table 438. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 438. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

Table 439. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Use this monitor element to determine the total volume of data sent over FCM through table queue buffer sends.

---

## fcm\_tq\_send\_wait\_time - FCM table queue send wait time monitor element

The time spent waiting to send the next buffer through a table queue. This reflects the time spent waiting for window count acknowledgements from the receiver end of the table queue. The value is given in milliseconds.

Table 440. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

Table 441. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Use this monitor element to determine how much time is being spent waiting to send a data buffer over FCM through a table queue.

## fcm\_tq\_sends\_total - FCM table queue send total monitor element

The total number of buffers containing table queue data sent using the internal FCM communications mechanism.

Table 442. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

Table 443. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Use this element in conjunction with **fc<sub>m</sub>\_tq\_send\_volume** and **fc<sub>m</sub>\_tq\_send\_wait\_time** monitor elements to determine the average data volume and time waited for buffer sent using a table queue.

---

## fetch\_count - Number of Successful Fetches

The number of successful physical fetches or the number of attempted physical fetches, depending on the snapshot monitoring level.

- For the stmt and dynsql snapshot monitoring levels and the statement event type: the number of successful fetches performed on a specific cursor.
- For the dcs\_stmt snapshot monitoring level: The number of attempted physical fetches during a statement's execution (regardless of how many rows were fetched by the application). In this situation, **fetch\_count** represents the number of times the server needed to send a reply data back to the gateway while processing a statement.

Table 444. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement
DCS Statement	dcs_stmt	Statement
Dynamic SQL	dynsql	Statement

For Dynamic SQL snapshot monitoring, this counter can be reset.

Table 445. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_stmt	-

## Usage

You can use this element to gain insight into the current level of activity within the database manager.

For performance reasons, a statement event monitor does not generate a statement event record for every FETCH statement. A record event is only generated when a FETCH returns a non-zero SQLCODE.

---

## files\_closed - Database files closed monitor element

The total number of database files closed.

Table 446. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 447. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 448. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-

## Usage

The database manager opens files for reading and writing into and out of the buffer pool. The maximum number of database files open by an application at any time is controlled by the **maxfilop** configuration parameter. If the maximum is reached, one file will be closed before the new file is opened. Note that the actual number of files opened may not equal the number of files closed.

You can use this element to help you determine the best value for the **maxfilop** configuration parameter.

---

## first\_active\_log - First Active Log File Number

The file number of the first active log file.

### Element identifier

first\_active\_log

### Element type

information

Table 449. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	detail_log	Basic

Table 450. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

**Usage** Use this element in conjunction with the *last\_active\_log* and *current\_active\_log* elements to determine the range of active log files. Knowing the range of active log files helps you determine the disk space required for log files.

You can also use this element to determine which log files have data to help you identify log files needed for split mirror support.

---

## first\_overflow\_time - Time of First Event Overflow

The date and time of the first overflow recorded by this overflow record.

Table 451. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Overflow Record	event_overflow	-

**Usage** Use this element with *last\_overflow\_time* to calculate the elapsed time for which the overflow record was generated.

---

## fs\_caching - File system caching monitor element

Indicates whether a particular table space uses file system caching. If **fs\_caching** is 0, file system caching is enabled. If **fs\_caching** is 1, file system caching is disabled.

Table 452. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 453. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table space	tablespace	Basic

Table 454. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Tablespaces	event_tablespace	-

---

## fs\_id - Unique file system identification number monitor element

This element shows the unique identification number provided by the operating system for a file system pointed to by a storage path or container.

Table 455. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE

Table 456. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	db_sto_path_info	Buffer Pool

### Usage

Use this element together with the following elements to gather data on space utilization for the database:

- **db\_storage\_path**

- `sto_path_free_sz`
- `fs_used_size`
- `fs_total_size`
- `fs_type`

---

## fs\_total\_size - Total size of a file system monitor element

This element shows the capacity of a file system pointed to by a storage path or container.

*Table 457. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE

*Table 458. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	db_sto_path_info	Buffer Pool

### Usage

You can use this element together with the following elements to gather data on space utilization for the database:

- `db_storage_path`
- `sto_path_free_sz`
- `fs_used_size`
- `fs_id`
- `fs_type`

---

## fs\_type - File System Type

This element shows the type of a file system that is pointed to by a storage path. This file system type is provided by the operating system..

**Element identifier**  
`fs_type`

**Element type**  
information

*Table 459. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	db_sto_path_info	Buffer Pool

**Usage** You can use this element together with the following elements to gather data on space utilization for the database:

- `db_storage_path`
- `sto_path_free_sz`
- `fs_used_size`
- `fs_total_size`



- fs\_id

---

## fs\_used\_size - Amount of space used on a file system monitor element

This element shows the amount of space already used on a file system pointed to by a storage path or container.

*Table 460. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE

*Table 461. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	db_sto_path_info	Buffer Pool

### Usage

You can use this element together with the following elements to gather data on space utilization for the database:

- db\_storage\_path
- sto\_path\_free\_sz
- fs\_total\_size
- fs\_id
- fs\_type

---

## gw\_comm\_error\_time - Communication Error Time

The date and time when the most recent communication error (SQL30081) occurred while a DCS application was attempting to connect to a host database, or while it was processing an SQL statement.

*Table 462. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Timestamp

### Usage

Use this element for problem determination, in conjunction with Communication Error and the communication error logged in administration notification log.

---

## gw\_comm\_errors - Communication Errors

The number of times that a communication error (SQL30081) occurred while a DCS application was attempting to connect to a host database, or while it was processing an SQL statement.

*Table 463. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Basic

For snapshot monitoring, this counter can be reset.

**Usage** By monitoring the number of communication errors over time, you can assess whether your DB2 Connect gateway has connectivity problems with a particular host database. You can establish what you consider to be a normal error threshold, so that any time the number of errors exceeds this threshold an investigation of the communication errors should be made.

Use this element for problem determination, in conjunction with the communication error logged in administration notification log.

---

## gw\_con\_time - DB2 Connect Gateway First Connect Initiated

The date and time when the first connection to the host database was initiated from the DB2 Connect gateway.

*Table 464. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dc_s_dbase	Timestamp
DCS Application	dc_s_appl	Timestamp

**Usage** Use this element for problem determination on DCS applications.

---

## gw\_connections\_top - Maximum Number of Concurrent Connections to Host Database

The maximum number of concurrent connections to a host database that have been handled by the DB2 Connect gateway since the first database connection.

**Element identifier**

gw\_connections\_top

**Element type**

watermark

*Table 465. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dc_s_dbase	Basic

**Usage** This element will help you understand the level of activity at the DB2 Connect gateway and the associated use of system resources.

---

## gw\_cons\_wait\_client - Number of Connections Waiting for the Client to Send Request

The current number of connections to host databases being handled by the DB2 Connect gateway that are waiting for the client to send a request.

**Element identifier**

gw\_cons\_wait\_client

**Element type**

gauge

Table 466. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic
DCS Database	dcs_dbase	Basic

**Usage** This value can change frequently. It should be sampled at regular intervals over an extended period in order to obtain a realistic view of gateway usage.

---

## gw\_cons\_wait\_host - Number of Connections Waiting for the Host to Reply

The current number of connections to host databases being handled by the DB2 Connect gateway that are waiting for a reply from the host.

**Element identifier**

gw\_cons\_wait\_host

**Element type**

gauge

Table 467. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic
DCS Database	dcs_dbase	Basic

**Usage** This value can change frequently. It should be sampled at regular intervals over an extended period in order to obtain a realistic view of gateway usage.

---

## gw\_cur\_cons - Current Number of Connections for DB2 Connect

The current number of connections to host databases being handled by the DB2 Connect gateway.

Table 468. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic
DCS Database	dcs_dbase	Basic

**Usage** This element will help you understand the level of activity at the DB2 Connect gateway and the associated use of system resources.

---

## gw\_db\_alias - Database Alias at the Gateway

The alias used at the DB2 Connect gateway to connect to the host database.

**Element identifier**

gw\_db\_alias

**Element type**

information

Table 469. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dc_s_appl_info	Basic

**Usage** Use this element for problem determination on DCS applications.

---

## gw\_exec\_time - Elapsed Time Spent on DB2 Connect Gateway Processing

The time in seconds and microseconds at the DB2 Connect gateway to process an application request (since the connection was established), or to process a single statement.

Table 470. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dc_s_appl	Statement, Timestamp
DCS Statement	dc_s_stmt	Statement, Timestamp

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to determine what portion of the overall processing time is due to DB2 Connect gateway processing.

---

## gw\_total\_cons - Total Number of Attempted Connections for DB2 Connect

The total number of connections attempted from the DB2 Connect gateway since the last db2start command or the last reset.

Table 471. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic
DCS Database	dc_s_dbase	Basic

For snapshot monitoring, this counter can be reset.

**Usage** This element will help you understand the level of activity at the DB2 Connect gateway and the associated use of system resources.

---

## hadr\_connect\_status - HADR Connection Status monitor element

The current high availability disaster recovery (HADR) connection status of the database.

Table 472. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

## Usage

Use this element to determine the HADR connection status of a database.

The data type of this element is integer.

If the database is in HADR primary or standby role, the value for this element is one of the following constants:

### SQLM\_HADR\_CONN\_CONNECTED

The database is connected to its partner node.

### SQLM\_HADR\_CONN\_DISCONNECTED

The database is not connected to its partner node.

### SQLM\_HADR\_CONN\_CONGESTED

The database is connected to its partner node, but the connection is congested. A connection is congested when the TCP/IP socket connection between the primary-standby pair is still alive, but one end cannot send to the other end. For example, the receiving end is not receiving from the socket connection, resulting in a full TCP/IP send space. The reasons for network connection being congested include the following:

- The network is being shared by too many resources or the network is not fast enough for the transaction volume of the primary HADR node.
- The server on which the standby HADR node resides is not powerful enough to retrieve information from the communication subsystem at the necessary rate.

This element should be ignored if the database's HADR role is standard. Use the **hadr\_role** monitor element to determine the HADR role of the database.

---

## hadr\_connect\_time - HADR Connection Time monitor element

Shows one of the following: high availability disaster recovery (HADR) connection time, HADR congestion time, or HADR disconnection time.

*Table 473. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

## Usage

Use this element to determine when the current HADR connection status began.

If the database is in HADR primary or standby role, the meaning of this element depends on the value of the **hadr\_connect\_status** element:

- If the value of the **hadr\_connect\_status** element is **SQLM\_HADR\_CONN\_CONNECTED**, then this element shows connection time.
- If the value of the **hadr\_connect\_status** element is **SQLM\_HADR\_CONN\_CONGESTED**, then this element shows the time when congestion began.
- If the value of the **hadr\_connect\_status** element is **SQLM\_HADR\_CONN\_DISCONNECTED**, then this element shows disconnection time.

If there has been no connection since the HADR engine dispatchable unit (EDU) was started, connection status is reported as Disconnected and HADR EDU startup time is used for the disconnection time. Since HADR connect and disconnect events are relatively infrequent, the time is collected and reported even if the DFT\_MON\_TIMESTAMP switch is off.

This element should be ignored if the database's HADR role is standard. Use the `hadr_role` monitor element to determine the HADR role of the database.

---

## hadr\_heartbeat - HADR Heartbeat monitor element

Number of missed heartbeats on the high availability disaster recovery (HADR) connection. If the database is in HADR primary or standby role, this element indicates the health of the HADR connection.

*Table 474. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

For snapshot monitoring, this counter cannot be reset.

### Usage note:

Use this element to determine the health of the HADR connection.

A heartbeat is a message sent from the other HADR database at regular intervals. If the value for this element is zero, no heartbeats have been missed and the connection is healthy. The higher the value, the worse the condition of the connection.

In disconnected mode, heartbeat missed is always shown as 0, because it is not applicable.

An HADR database expects at least one heartbeat message from the other database in each quarter of the time interval defined in the HADR\_TIMEOUT database configuration parameter, or in 30 seconds, whichever is shorter. For example, if the HADR\_TIMEOUT value is 80 (seconds), then the HADR database expects at least one heartbeat message from the other database every 20 seconds.

The data type of this element is integer.

This element should be ignored if the database's HADR role is standard. Use the `hadr_role` monitor element to determine the HADR role of the database.

---

## hadr\_local\_host - HADR Local Host monitor element

The local high availability disaster recovery (HADR) host name. The value is displayed as a host name string or an IP address string such as "1.2.3.4".

*Table 475. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

## Usage

Use this element to determine the effective HADR local host name. HADR database configuration parameters are static. Changes to a parameter are not effective until the database is stopped and restarted. This monitor element reports the value that the HADR system is actually using rather than the value in the database configuration file.

This element should be ignored if the database's HADR role is standard. Use the **hadr\_role** monitor element to determine the HADR role of the database.

**Note:** Any name used must resolve to one IP address. A name that resolves to more than one address will cause an error when trying to start HADR.

---

## hadr\_local\_service - HADR Local Service monitor element

The local HADR TCP service. This value is displayed as a service name string or a port number string.

*Table 476. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

## Usage

Use this element to determine the effective HADR local service name. HADR database configuration parameters are static. Changes to a parameter are not effective until the database is stopped and restarted. This monitor element reports the value the HADR system is actually using rather than the value in the database configuration file.

This element should be ignored if the database's HADR role is standard. Use the **hadr\_role** monitor element to determine the HADR role of the database.

---

## hadr\_log\_gap - HADR Log Gap

This element shows the running average of the gap between the primary Log sequence number (LSN) and the standby log LSN. The gap is measured in number of bytes.

*Table 477. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

## Usage

Use this element to determine the gap between the primary and standby HADR database logs.

When a log file is truncated, the LSN in the next log file starts as if the last file were not truncated. This LSN hole does not contain any log data. Such holes can cause the log gap not to reflect the actual log difference between the primary and the standby HADR database logs.

This element should be ignored if the database's HADR role is standard. Use the `hadr_role` monitor element to determine the HADR role of the database.

---

## hadr\_peer\_window - HADR peer window monitor element

The value of the `HADR_PEER_WINDOW` database configuration parameter.

*Table 478. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

### Usage

Use this element to determine the value of the `HADR_PEER_WINDOW` database configuration parameter.

---

## hadr\_peer\_window\_end - HADR peer window end monitor element

The point in time until which a high availability disaster recovery (HADR) primary database promises to stay in peer or disconnected peer state, as long as the primary database is active.

*Table 479. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

### Usage

Use this element to determine the point in time until which the primary promises to stay in peer or disconnected peer state.

The value reported by the primary database might be different from the value reported by the standby database. This occurs because the primary database updates the value when it sends a heartbeat message, but the new value is shown on the standby database only after the message is received and processed on the standby database.

If a database moves out of peer or disconnected peer state, the value of this monitor element is not reset. The last known value is kept and returned. If a database never reached peer state, a value of zero will be returned.

The peer window end time is set by the primary database and then sent to the standby database. For this reason, the value of the peer window end is based on the clock of the primary database. When you compare the peer window end time with the primary database down time, you might need to add an offset to convert the timestamp to the primary database clock, if the two clocks are not well synchronized.

---

## hadr\_primary\_log\_file - HADR Primary Log File monitor element

The name of the current log file on the primary HADR database.



Table 480. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

## Usage

Use this element to determine the current log file on the primary HADR database.

This element should be ignored if the database's HADR role is standard. Use the **hadr\_role** monitor element to determine the HADR role of the database.

---

## hadr\_primary\_log\_lsn - HADR Primary Log LSN monitor element

The current log position of the primary HADR database. Log sequence number (LSN) is a byte offset in the database's log stream.

Table 481. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

## Usage

Use this element to determine the current log position on the primary HADR database.

This element should be ignored if the database's HADR role is standard. Use the **hadr\_role** monitor element to determine the HADR role of the database.

---

## hadr\_primary\_log\_page - HADR Primary Log Page monitor element

The page number in the current log file indicating the current log position on the primary HADR database. The page number is relative to the log file. For example, page zero is the beginning of the file.

Table 482. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

## Usage

Use this element to determine the current log page on the primary HADR database.

This element should be ignored if the database's HADR role is standard. Use the **hadr\_role** monitor element to determine the HADR role of the database.

---

## hadr\_remote\_host - HADR Remote Host monitor element

The remote high availability disaster recovery (HADR) host name. The value is displayed as a host name string or an IP address string such as "1.2.3.4".

Table 483. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

## Usage

Use this element to determine the effective HADR remote host name. HADR database configuration parameters are static. Changes to a parameter are not effective until the database is stopped and restarted. This monitor element reports the value the HADR system is actually using rather than the value in the database configuration file.

This element should be ignored if the database's HADR role is standard. Use the *hadr\_role* monitor element to determine the HADR role of the database.

**Note:** Any name used must resolve to one IP address. A name that resolves to more than one address will cause an error when trying to start HADR.

---

## hadr\_remote\_instance - HADR Remote Instance monitor element

The remote HADR instance name.

Table 484. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

## Usage

Use this element to determine the effective HADR remote instance name. HADR database configuration parameters are static. Changes to a parameter are not effective until the database is stopped and restarted. This monitor element reports the value the HADR system is actually using rather than the value in the database configuration file.

This element should be ignored if the database's HADR role is standard. Use the *hadr\_role* monitor element to determine the HADR role of the database.

---

## hadr\_remote\_service - HADR Remote Service monitor element

The remote HADR TCP service. This value is displayed as a service name string or a port number string.

Table 485. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

## Usage

Use this element to determine the effective HADR remote service name. HADR database configuration parameters are static. Changes to a parameter are not

effective until the database is stopped and restarted. This monitor element reports the value the HADR system is actually using rather than the value in the database configuration file.

This element should be ignored if the database's HADR role is standard. Use the `hadr_role` monitor element to determine the HADR role of the database.

---

## hadr\_role - HADR Role

The current high availability disaster recovery (HADR) role of the database.

*Table 486. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

### Usage

Use this element to determine the HADR role of a database.

The data type of this element is integer.

The value for this element is one of the following constants:

**SQLM\_HADR\_ROLE\_STANDARD**

The database is not an HADR database.

**SQLM\_HADR\_ROLE\_PRIMARY**

The database is the primary HADR database.

**SQLM\_HADR\_ROLE\_STANDBY**

The database is the standby HADR database.

---

## hadr\_standby\_log\_file - HADR Standby Log File monitor element

The name of the current log file on the standby HADR database.

*Table 487. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

### Usage

Use this element to determine the current log file on the standby HADR database.

This element should be ignored if the database's HADR role is standard. Use the `hadr_role` monitor element to determine the HADR role of the database.

---

## hadr\_standby\_log\_lsn - HADR Standby Log LSN monitor element

The current log position of the standby HADR database. Log sequence number (LSN) is a byte offset in the database's log stream.

Table 488. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

## Usage

Use this element to determine the current log position on the standby HADR database.

This element should be ignored if the database's HADR role is standard. Use the **hadr\_role** monitor element to determine the HADR role of the database.

---

## hadr\_standby\_log\_page - HADR Standby Log Page monitor element

The page number in the current log file indicating the current log position on the standby HADR database. The page number is relative to the log file. For example, page zero is the beginning of the file.

Table 489. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

## Usage

Use this element to determine the current log page on the standby HADR database.

This element should be ignored if the database's HADR role is standard. Use the **hadr\_role** monitor element to determine the HADR role of the database.

---

## hadr\_state - HADR State monitor element

The current high availability disaster recovery (HADR) state of the database.

Table 490. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

## Usage

Use this element to determine the HADR state of a database.

The data type of this element is integer. If the database is in HADR primary or standby role, the value for this element is one of the following constants:

### **SQLM\_HADR\_STATE\_DISCONNECTED**

The database is not connected to its partner database.

### **SQLM\_HADR\_STATE\_LOC\_CATCHUP**

The database is doing local catchup.

### **SQLM\_HADR\_STATE\_REM\_CATCH\_PEND**

The database is waiting to connect to its partner to do remote catchup.

**SQLM\_HADR\_STATE\_REM\_CATCHUP**

The database is doing remote catchup.

**SQLM\_HADR\_STATE\_PEER**

The primary and standby databases are connected and are in peer state.

**SQLM\_HADR\_STATE\_DISCONN\_PEER**

The primary and standby databases are in disconnected peer state.

This element should be ignored if the database's HADR role is standard. Use the **hadr\_role** monitor element to determine the HADR role of the database.

---

**hadr\_syncmode - HADR Synchronization Mode monitor element**

The current high availability disaster recovery (HADR) synchronization mode of the database.

*Table 491. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

**Usage**

Use this element to determine the HADR synchronization mode of a database.

The data type of this element is integer.

HADR database configuration parameters are static. Changes to a parameter are not effective until the database is stopped and restarted. This monitor element reports the value the HADR system is actually using rather than the value in the database configuration file.

If the database is in HADR primary or standby role, the value for this element is one of the following constants:

**SQLM\_HADR\_SYNCMODE\_SYNC**

Sync mode.

**SQLM\_HADR\_SYNCMODE\_NEARSYNC**

Nearsync mode.

**SQLM\_HADR\_SYNCMODE\_ASYNC**

Async mode.

This element should be ignored if the database's HADR role is standard. Use the **hadr\_role** monitor element to determine the HADR role of the database.

---

**hadr\_timeout - HADR Timeout monitor element**

The number of seconds without any communication from its partner after which an HADR database server will consider that the connection between them has failed.

*Table 492. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	hadr	Basic

## Usage

Use this element to determine the effective HADR timeout value. HADR database configuration parameters are static. Changes to a parameter are not effective until the database is stopped and restarted. This monitor element reports the value the HADR system is actually using rather than the value in the database configuration file.

This element should be ignored if the database's HADR role is standard. Use the `hadr_role` monitor element to determine the HADR role of the database.

---

## hash\_join\_overflows - Hash Join Overflows

The number of times that hash join data exceeded the available sort heap space.

### Element identifier

hash\_join\_overflows

### Element type

counter

*Table 493. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

*Table 494. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** At the database level, if the value of `hash_join_small_overflows` is greater than 10% of this `hash_join_overflows`, then you should consider increasing the sort heap size. Values at the application level can be used to evaluate hash join performance for individual applications.

---

## hash\_join\_small\_overflows - Hash Join Small Overflows

The number of times that hash join data exceeded the available sort heap space by less than 10%.

### Element identifier

hash\_join\_small\_overflows

### Element type

counter

*Table 495. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 496. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** If this value and `hash_join_overflows` are high, then you should consider increasing the sort heap threshold. If this value is greater than 10% of `hash_join_overflows`, then you should consider increasing the sort heap size.

---

## histogram\_type - Histogram type monitor element

The type of the histogram, in string format.

There are six histogram types.

### **CoordActQueueTime**

A histogram of the time non-nested activities spend queued (for example, in a threshold queue), measured on the coordinator partition.

### **CoordActExecTime**

A histogram of the time non-nested activities spend executing at the coordinator partition. Execution time does not include time spent initializing or queued. For cursors, execution time includes only the time spent on open, fetch and close requests. When an activity is remapped between service subclasses, the execution time histogram is updated only for the service subclass in which the activity completes execution.

### **CoordActLifetime**

A histogram of the elapsed time from when a non-nested activity is identified by the database manager until the activity completes execution, as measured on the coordinator partition. When you remap activities between service subclasses, the lifetime histogram is updated only for the service subclass in which the activity completes execution.

### **CoordActInterArrivalTime**

A histogram of the time interval between the arrival of non-nested coordinator activities. The inter-arrival time mean is calculated for service subclasses through which activities enter the system. When you remap activities between service subclasses, the inter-arrival time histogram of the service subclass you remap an activity to is unaffected.

### **CoordActEstCost**

A histogram of the estimated cost of non-nested DML activities. The estimated cost of an activity is counted only towards the service subclass in which the activity enters the system.

### **ReqExecTime**

A histogram of request execution times, which includes requests on the coordinator partition, and any subrequests on both coordinator and non-coordinator partitions (like RPC requests or SMP subagent requests). Requests included may or may not be associated with an activity: Both PREPARE and OPEN requests are included in this histogram, for example, but while OPEN requests are always associated with a cursor activity, PREPARE requests are not part of any activity. The execution time

histogram of a service subclass involved in remapping counts the portion of the execution time spent by the partial request in the service subclass.

Table 497. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_histogrambin	-

## Usage

Use this element to identify the type of histogram. Several histograms can belong to the same statistics record, but only one of each type.

---

## host\_ccsid - Host Coded Character Set ID

This is the coded character set identifier (CCSID) of the host database.

**Element identifier**  
host\_ccsid

**Element type**  
information

Table 498. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcx_appl_info	Basic

**Usage** Use this element for problem determination on DCS applications.

---

## host\_db\_name - Host Database Name

The real name of the host database for which information is being collected or to which the application is connected. This is the name that was given to the database when it was created.

Table 499. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcx_dbase	Basic
DCS Application	dcx_appl_info	Basic

**Usage** Use this element for problem determination on DCS applications.

---

## host\_prdid - Host Product/Version ID

The product and version that is running on the server.

Table 500. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcx_appl_info	Basic

**Usage** Used to identify the product and code version of the DRDA host database product. It is in the form PPPVRRM, where:

- PPP identifies the host DRDA product
  - ARI for DB2 Server for VSE & VM



- DSN for DB2 for z/OS
- QSQ for DB2 for i
- SQL for other DB2 products.
- VV identifies a 2-digit version number (with high-order 0 in the case of a 1-digit version)
- RR identifies a 2-digit release number (with high-order 0 in the case of a 1-digit release)
- M identifies a 1-character modification level (0-9 or A-Z)

---

## host\_response\_time - Host Response Time

At the DCS statement level, this is the elapsed time between the time that the statement was sent from the DB2 Connect gateway to the host for processing and the time when the result was received from the host. At DCS database and DCS application levels, it is the sum of the elapsed times for all the statements that were executed for a particular application or database. At the data transmission level, this is the sum of host response times for all the statements that used this many data transmissions.

*Table 501. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dc_s_dbase	Statement
DCS Application	dc_s_appl	Statement, Timestamp
DCS Statement	dc_s_stmt	Statement, Timestamp
Data Transmission	stmt_transmissions	Statement, Timestamp

For snapshot monitoring at the statement level, this counter cannot be reset. This counter can be reset at other levels.

**Usage** Use this element with Outbound Number of Bytes Sent and Outbound Number of Bytes Received to calculate the outbound response time (transfer rate):

$$(\text{outbound bytes sent} + \text{outbound bytes received}) / \text{host response time}$$

---

## idle\_agents - Number of Idle Agents

The number of agents in the agent pool that are currently unassigned to an application and are, therefore, "idle".

*Table 502. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

**Usage** You can use this element to help set the *num\_poolagents* configuration parameter. Having idle agents available to service requests for agents can improve performance.

---

## iid - Index identifier monitor element

Identifier for the index.

Table 503. Table Function Monitoring Information

Table Function	Monitor Element	Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected	

## inbound\_bytes\_received - Inbound Number of Bytes Received

The number of bytes received by the DB2 Connect gateway from the client, excluding communication protocol overhead (for example, TCP/IP or SNA headers).

Table 504. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcs_appl	Basic
DCS Statement	dcs_stmt	Statement

For snapshot monitoring at the application level, this counter can be reset. This counter cannot be reset at other levels.

**Usage** Use this element to measure the throughput from the client to the DB2 Connect gateway.

## inbound\_bytes\_sent - Inbound Number of Bytes Sent

The number of bytes sent by the DB2 Connect gateway to the client, excluding communication protocol overhead (for example, TCP/IP or SNA headers).

Table 505. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcs_appl	Basic
DCS Statement	dcs_stmt	Statement

For snapshot monitoring at the application level, this counter can be reset. This counter cannot be reset at other levels.

**Usage** Use this element to measure the throughput from the DB2 Connect gateway to the client.

## inbound\_comm\_address - Inbound Communication Address

This is the communication address of the client. For example, it could be an SNA net ID and LU partner name, or an IP address and port number for TCP/IP.

Table 506. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
DCS Application	dcs_appl_info	Basic

**Usage** Use this element for problem determination on DCS applications.

---

## include\_col\_updates - Include column updates monitor element

Number of include column updates.

*Table 507. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

---

---

## index\_object\_pages - Index Object Pages

The number of disk pages consumed by all indexes defined on a table.

*Table 508. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table	Basic

---

*Table 509. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Tables	event_table	-

---

**Usage** This element provides a mechanism for viewing the actual amount of space consumed by indexes defined on a particular table. This element can be used in conjunction with a table event monitor to track the rate of index growth over time. This element is not returned for partitioned tables.

---

## index\_only\_scans - Index-only scans monitor element

Number of index-only scans. Index-only scans occur when the results of scan was satisfied by access to the index only.

*Table 510. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

---

---

## index\_scans - Index scans monitor element

Number of index scans.

*Table 511. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

---

---

## index\_tbsp\_id - Index table space ID monitor element

An identifier of the table space that holds indexes created on this table.

Table 512. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLE table function - Get table metrics	Always collected

## Usage

The value of this element matches a value from column TBSPACEID of view SYSCAT.TABLESPACES.

---

## input\_db\_alias - Input Database Alias

The alias of the database provided when calling the snapshot function.

Table 513. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl_id_info	Basic
Table Space	tablespace_list	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Table	table_list	Table
Lock	db_lock_list	Basic

**Usage** This element can be used to identify the specific database to which the monitor data applies. It contains blanks unless you requested monitor information related to a specific database.

The value of this field may be different than the value of the *client\_db\_alias* monitor element since a database can have many different aliases. Different applications and users can use different aliases to connect to the same database.

---

## insert\_sql\_stmts - Inserts

This element contains a count of the total number of times the federated server has issued an INSERT statement to this data source on behalf of any application since the start of the federated server instance or the last reset of the database monitor counters, whichever is latest.

Table 514. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

## Usage

Use this element to determine the level of database activity directed against this data source by the federated server or an application.

You can also use this element to determine the percentage of write activity against this data source by the federated server or an application, with the following formula:

$$\text{write\_activity} = \frac{(\text{INSERT statements} + \text{UPDATE statements} + \text{DELETE statements})}{(\text{SELECT statements} + \text{INSERT statements} + \text{UPDATE statements} + \text{DELETE statements})}$$


---

## insert\_time - Insert Response Time

This element contains the aggregate amount of time, in milliseconds, that it has taken this data source to respond to INSERTs from all applications or a single application running on this federated server instance since the start of the federated server instance or the last reset of the database monitor counters, whichever is latest.

The response time is measured as the difference in time between the time the federated server submits an INSERT statement to the data source, and the time the data source responds to the federated server, indicating that the INSERT has been processed.

*Table 515. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Timestamp
Application	appl_remote	Timestamp

For snapshot monitoring, this counter can be reset.

### Usage

Use this element to determine the actual amount of time that transpires waiting for INSERTs to this data source to be processed. This information can be useful for capacity planning and tuning.

---

## insert\_timestamp - Statement insert timestamp monitor element

The time when the statement was inserted into the cache.

*Table 516. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

*Table 517. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Dynamic SQL	dynsql	Basic

### Usage

This element specifies the time when the statement was inserted into the cache. It can be used to estimate the lifetime of a statement in the cache.

---

## int\_auto\_rebinds - Internal Automatic Rebinds

The number of automatic rebinds (or recompiles) that have been attempted.

*Table 518. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

*Table 519. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** Automatic rebinds are the internal binds the system performs when a package has been invalidated. The rebind is performed the first time that the database manager needs to execute an SQL statement from the package. For example, packages are invalidated when you:

- Drop an object, such as a table, view, or index, on which the plan is dependent
- Add or drop a foreign key
- Revoke object privileges on which the plan is dependent.

You can use this element to determine the level of database activity at the application or database level. Since `int_auto_rebinds` can have a significant impact on performance, they should be minimized where possible.

You can also use this element to determine the percentage of rebind activity using the following formula:

$$\text{int\_auto\_rebinds} / \text{total number of statements}$$

This information can be useful for analyzing application activity and throughput.

---

## int\_commits - Internal Commits

The total number of commits initiated internally by the database manager.

*Table 520. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

*Table 521. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** An internal commit may occur during any of the following:

- A reorganization
- An import
- A bind or pre-compile
- An application ends without executing an explicit SQL COMMIT statement (on UNIX).

This value, which does not include explicit SQL COMMIT statements, represents the number of these internal commits since the later of:

- The connection to the database (for database-level information, this is the time of the first connection)
- The last reset of the database monitor counters.

You can use this element to calculate the total number of units of work by calculating the sum of the following:

```
commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks
```

**Note:** The units of work calculated will only include those since the later of:

- The connection to the database (for database-level information, this is the time of the first connection)
- The last reset of the database monitor counters.

This calculation can be done at the application or the database level.

---

## int\_deadlock\_rollbacks - Internal Rollbacks Due To Deadlock

The total number of forced rollbacks initiated by the database manager due to a deadlock. A rollback is performed on the current unit of work in an application selected by the database manager to resolve the deadlock.

**Element identifier**

int\_deadlock\_rollbacks

**Element type**

counter

*Table 522. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

*Table 523. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

**Usage** This element shows the number of deadlocks that have been broken and

can be used as an indicator of concurrency problems. It is important, since `int_deadlock_rollbacks` lower the throughput of the database.

This value is included in the value given by `int_rollbacks`.

---

## int\_node\_splits - Intermediate node splits monitor element

Number of times an intermediate index node was split during an insert operation.

*Table 524. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

---

## int\_rollbacks - Internal Rollbacks

The total number of rollbacks initiated internally by the database manager.

### Element identifier

`int_rollbacks`

### Element type

counter

*Table 525. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

*Table 526. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** An internal rollback occurs when any of the following **cannot** complete successfully:

- A reorganization
- An import
- A bind or pre-compile
- An application ends as a result of a deadlock situation or lock timeout situation
- An application ends without executing an explicit commit or rollback statement (on Windows).

This value represents the number of these internal rollbacks since the later of:

- The connection to the database (for database-level information, this is the time of the first connection)
- The last reset of the database monitor counters.



While this value does not include explicit SQL ROLLBACK statements, the count from `int_deadlock_rollbacks` is included.

You can use this element to calculate the total number of units of work by calculating the sum of the following:

```
commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks
```

**Note:** The units of work calculated will include those since the later of:

- The connection to the database (for database-level information, this is the time of the first connection)
- The last reset of the database monitor counters.

This calculation can be done at the application or the database level.

---

## int\_rows\_deleted - Internal Rows Deleted

This is the number of rows deleted from the database as a result of internal activity.

*Table 527. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic
Application	stmt	Basic
Dynamic SQL	dynsql	Statement

For snapshot monitoring, this counter can be reset.

*Table 528. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-
Statements	event_stmt	-

**Usage** This element can help you gain insight into internal activity within the database manager of which you might not be aware. If this activity is high, you may want to evaluate your table design to determine if the referential constraints or triggers that you have defined on your database are necessary.

Internal delete activity can be a result of:

- A cascading delete enforcing an ON CASCADE DELETE referential constraint
- A trigger being fired.

---

## int\_rows\_inserted - Internal Rows Inserted

The number of rows inserted into the database as a result of internal activity caused by triggers.

Table 529. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic
Application	stmt	Basic
Dynamic SQL	dynsql	Statement

For snapshot monitoring, this counter can be reset.

Table 530. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-
Statements	event_stmt	-

**Usage** This element can help you gain insight into the internal activity within the database manager. If this activity is high, you may want to evaluate your design to determine if you can alter it to reduce this activity.

---

## int\_rows\_updated - Internal Rows Updated

This is the number of rows updated from the database as a result of internal activity.

Table 531. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic
Application	stmt	Basic
Dynamic SQL	dynsql	Statement

For snapshot monitoring, this counter can be reset.

Table 532. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-
Statements	event_stmt	-

**Usage** This element can help you gain insight into internal activity within the database manager of which you might not be aware. If this activity is high, you may want to evaluate your table design to determine if the referential constraints that you have defined on your database are necessary.

Internal update activity can be a result of:

- A *set null* row update enforcing a referential constraint defined with the ON DELETE SET NULL rule
- A trigger being fired.

---

## invocation\_id - Invocation ID monitor element

An identifier that distinguishes one particular invocation of this activity from others at the same nesting level.

*Table 533. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### Usage

---

## ipc\_rcv\_volume - Interprocess communication received volume monitor element

The amount of data received by data server from clients over IPC. This value is reported in bytes.

*Table 534. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

*Table 535. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE

Table 535. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	Reported in the in the system_metrics document.	-

## ipc\_rcv\_wait\_time - Interprocess communication received wait time monitor element

The time spent by an agent receiving an incoming client request using the IPC communications protocol. The value is reported in milliseconds.

Table 536. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 537. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## ipc\_rcvs\_total - Interprocess communication receives total monitor element

The number of times data was received by the database server from the client application using IPC.

Table 538. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 539. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## ipc\_send\_volume - Interprocess communication send volume monitor element

The amount of data sent by data server to clients over the IPC protocol. This value is reported in bytes.

Table 540. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 540. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 541. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## ipc\_send\_wait\_time - Interprocess communication send wait time monitor element

The time spent blocking on an IPC send to the client. The value is given in milliseconds.

Table 542. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 542. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 543. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## ipc\_sends\_total - Interprocess communication send total monitor element

The number of times data was sent by the database server to client applications using IPC.

Table 544. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE

Table 544. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 545. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## is\_system\_appl - Is System Application monitor element

Indicates whether the application is a system application.

Table 546. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic

### Usage

The **is\_system\_appl** monitor element indicates whether an application is an internal system application. The possible values are

- 0 user application
- 1 system application

## key\_updates - Key updates monitor element

Number of key updates.

Table 547. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

## last\_active\_log - Last Active Log File Number

The file number of the last active log file.

**Element identifier**

last\_active\_log

**Element type**

information



Table 548. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	detail_log	Basic

Table 549. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

**Usage** Use this element in conjunction with the *first\_active\_log* and *current\_active\_log* elements to determine the range of active log files. Knowing the range of active log files helps you determine the disk space required for log files.

You can also use this element to determine which log files have data to help you identify log files needed for split mirror support.

---

## last\_backup - Last Backup Timestamp

The date and time that the latest database backup was completed.

**Element identifier**  
last\_backup

**Element type**  
timestamp

Table 550. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Timestamp

**Usage** You may use this element to help you identify a database that has not been backed up recently, or to identify which database backup file is the most recent. If the database has never been backed up, this timestamp is initialized to zero.

---

## last\_extent - Last extent moved monitor element

The numeric identifier of the last extent moved by the table space rebalancer process.

Table 551. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_EXTENT_MOVEMENT_STATUS - Get extent movement progress status metrics	Always collected

---

## last\_overflow\_time - Time of Last Event Overflow

The date and time of the last overflow recorded this overflow record.

Table 552. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Overflow Record	event_overflow	-

**Usage** Use this element with *first\_overflow\_time* to calculate the elapsed time for which the overflow record was generated.

---

## last\_reference\_time - Last reference time monitor element

The last time the activity was accessed by a request.

*Table 553. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### Usage

---

## last\_reset - Last Reset Timestamp

Indicates the date and time that the monitor counters were reset for the application issuing the GET SNAPSHOT.

**Element identifier**  
last\_reset

**Element type**  
timestamp

*Table 554. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Timestamp
Database	dbase	Timestamp
Application	appl	Timestamp
Table Space	tablespace_list	Buffer Pool, Timestamp
Table	table_list	Timestamp
DCS Database	dcs_dbase	Timestamp
DCS Application	dcs_appl	Timestamp

**Usage** You can use this element to help you determine the scope of information returned by the database system monitor.

If the counters have never been reset, this element will be zero.

The database manager counters will only be reset if you reset all active databases.

---

## last\_wlm\_reset - Time of last reset monitor element

This element, in the form of a local timestamp, shows the time at which the last statistics event record of this type was created.

*Table 555. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-

Table 555. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wlstats	-
Statistics	event_wcstats	-
Statistics	event_qstats	-

## Usage

Use the **wlm\_last\_reset** and **statistics\_timestamp** monitor elements to determine a period of time over which the statistics in an event monitor statistics record were collected. The collection interval begins at the **wlm\_last\_reset** time and ends at **statistics\_timestamp**.

---

## lob\_object\_pages - LOB Object Pages

The number of disk pages consumed by LOB data.

Table 556. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table	Basic

Table 557. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Tables	event_table	-

**Usage** This element provides a mechanism for viewing the actual amount of space consumed by LOB data in a particular table. This element can be used in conjunction with a table event monitor to track the rate of LOB data growth over time.

---

## local\_cons - Local Connections

The number of local applications that are currently connected to a database within the database manager instance being monitored.

Table 558. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

## Usage

This number can help you determine the level of concurrent processing occurring in the database manager. This value will change frequently, so you may need to sample it at specific intervals over an extended period of time to get a realistic view of system usage.

This number only includes applications that were initiated from the same instance as the database manager. The applications are connected, but may or may not be executing a unit of work in the database.

When used in conjunction with the `rem_cons_in` monitor element, this element can help you adjust the setting of the `max_connections` configuration parameter.

---

## local\_cons\_in\_exec - Local Connections Executing in the Database Manager

The number of local applications that are currently connected to a database within the database manager instance being monitored and are currently processing a unit of work.

Table 559. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

### Usage

This number can help you determine the level of concurrent processing occurring in the database manager. This value will change frequently, so you may need to sample it at specific intervals over an extended period of time to get a realistic view of system usage. This number only includes applications that were initiated from the same instance as the database manager.

When used in conjunction with the `rem_cons_in_exec` monitor element, this element can help you adjust the setting of the `max_coordagents` configuration parameter.

The following recommendations apply to non-concentrator configurations only. When concentrator is enabled, DB2 is multiplexing a larger number of client connections onto a smaller pool of coordinator agents. In this case, it is usually acceptable to have the sum of `rem_cons_in_exec` and `local_cons_in_exec` approach the `max_coordagents` value.

- If `max_coordagents` is set to `AUTOMATIC`, do not make any adjustments.
- If `max_coordagents` is not set to `AUTOMATIC` and if the sum of `rem_cons_in_exec` and `local_cons_in_exec` is close to `max_coordagents`, increase the value of `max_coordagents`.

---

## local\_start\_time - Local start time monitor element

The time that this activity began doing work on the partition. It is in local time. This field can be an empty string when an activity has entered the system but is in a queue and has not started executing.

Table 560. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### Usage

---

## lock\_attributes - Lock attributes monitor element

Lock attributes.

Table 561. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Lock	lock	Basic
Lock	lock_wait	Basic

Table 562. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks <sup>1</sup>	lock	-
Deadlocks <sup>1</sup>	event_dlconn	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

## Usage

The following are possible lock attribute settings. Each lock attribute setting is based upon a bit flag value defined in sqlmon.h.

API Constant	Description
SQLM_LOCKATTR_WAIT_FOR_AVAIL	Wait for availability.
SQLM_LOCKATTR_ESCALATED	Acquired by escalation.
SQLM_LOCKATTR_RR_IN_BLOCK	RR lock "in" block.
SQLM_LOCKATTR_INSERT	Insert lock.
SQLM_LOCKATTR_DELETE_IN_BLOCK	Deleted row "in" block.
SQLM_LOCKATTR_RR	Lock by RR scan.
SQLM_LOCKATTR_UPDATE_DELETE	Update/delete row lock.
SQLM_LOCKATTR_ALLOW_NEW	Allow new lock requests.
SQLM_LOCKATTR_NEW_REQUEST	A new lock requestor.
SQLM_LOCKATTR_INDOUBT	Lock held by Indoubt Transaction.
SQLM_LOCKATTR_LOW_PRIORITY	Lock held by low priority application.

## lock\_count - Lock count monitor element

The number of locks on the lock being held.

Table 563. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Lock	lock	Basic

Table 564. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-

Table 564. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks <sup>1</sup>	lock	-
Deadlocks <sup>1</sup>	event_dlconn	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

## Usage

This value ranges from 1 to 255. It is incremented as new locks are acquired, and decremented as locks are released.

When the **lock\_count** monitor element has a value of 255, this indicates that a transaction duration lock is being held. At this point, the **lock\_count** monitor element is no longer incremented or decremented when locks are acquired or released. The **lock\_count** monitor element is set to a value of 255 in one of two possible ways:

1. The **lock\_count** monitor element value is incremented 255 times due to new locks being acquired.
2. A transaction duration lock is explicitly acquired. For example, with a LOCK TABLE statement, or an INSERT.

---

## lock\_current\_mode - Original Lock Mode Before Conversion

During a lock conversion operation, the type of lock held before the conversion is completed.

Table 565. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Lock	lock	Basic
Lock	lock_wait	Basic

Table 566. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks <sup>1</sup>	lock	-
Deadlocks <sup>1</sup>	event_dlconn	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

## Usage

The following scenario describes an example of lock conversion. During an update or delete operation it is possible to wait for an X lock on the target row. If the transaction is holding an S or V lock on the row, this would require a conversion.

At this point, the `lock_current_mode` element is assigned a value of S or V, while the lock waits to be converted to an X lock.

---

## lock\_escalation - Lock escalation monitor element

Indicates whether a lock request was made as part of a lock escalation.

*Table 567. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Lock	lock	Lock
Lock	lock_wait	Lock

*Table 568. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks <sup>1</sup>	lock	-
Deadlocks <sup>1</sup>	event_dlconn	-
Deadlocks with Details <sup>1</sup>	event_detailed_dlconn	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

### Usage

Use this element to better understand the cause of deadlocks. If you experience a deadlock that involves applications doing lock escalation, you may want to increase the amount of lock memory or change the percentage of locks that any one application can request.

---

## lock\_escals - Number of lock escalations monitor element

The number of times that locks have been escalated from several row locks to a table lock.

*Table 569. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 569. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 570. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 571. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Connection	event_conn	-
Transactions	event_xact	-

**Usage** A lock is escalated when the total number of locks held by an application reaches the maximum amount of lock list space available to the application, or the lock list space consumed by all applications is approaching the total lock list space. The amount of lock list space available is determined by the **maxlocks** and **locklist** configuration parameters.

When an application reaches the maximum number of locks allowed and there are no more locks to escalate, it will then use space in the lock list allocated for other applications. When the entire lock list is full, an error occurs.



This data item includes a count of all lock escalations, including exclusive lock escalations.

There are several possible causes for excessive lock escalations:

- The lock list size (**locklist**) may be too small for the number of concurrent applications
- The percent of the lock list usable by each application (**maxlocks**) may be too small
- One or more applications may be using an excessive number of locks.

To resolve these problems, you may be able to:

- Increase the **locklist** configuration parameter value.
- Increase the **maxlocks** configuration parameter value.
- Identify the applications with large numbers of locks (see the **locks\_held\_top** monitor element), or those that are holding too much of the lock list, using the following formula:

$$(((locks\ held * 36) / (locklist * 4096)) * 100)$$

and comparing the value to **maxlocks**. These applications can also cause lock escalations in other applications by using too large a portion of the lock list. These applications may need to resort to using table locks instead of row locks, although table locks may cause an increase in the values of **lock\_waits** and **lock\_wait\_time** monitor element.

## lock\_hold\_count - Lock hold count monitor element

The number of holds placed on the lock. Holds are placed on locks by cursors registered with the WITH HOLD clause and some DB2 utilities. Locks with holds are not released when transactions are committed.

Table 572. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Lock	lock	Basic

Table 573. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks <sup>1</sup>	lock	-
Deadlocks <sup>1</sup>	event_dlconn	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

## lock\_list\_in\_use - Total Lock List Memory In Use

The total amount of lock list memory (in bytes) that is in use.

**Element identifier**

lock\_list\_in\_use

**Element type**

watermark

Table 574. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

**Usage** This element may be used in conjunction with the *locklist* configuration parameter to calculate the lock list utilization. If the lock list utilization is high, you may want to consider increasing the size of that parameter.

**Note:** When calculating utilization, it is important to note that the *locklist* configuration parameter is allocated in pages of 4K bytes each, while this monitor element provides results in bytes.

## lock\_mode - Lock mode monitor element

The type of lock being held.

Table 575. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Lock
Lock	lock	Lock
Lock	lock_wait	Lock

Table 576. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks <sup>1</sup>	lock	-
Deadlocks <sup>1</sup>	event_dlconn	-
Deadlocks with Details <sup>1</sup>	event_detailed_dlconn	-

**1** This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

### Usage

This mode can help you determine the source of contention for resources.

This element indicates one of the following, depending on the type of monitor information being examined:

- The type of lock another application holds on the object that this application is waiting to lock (for application-monitoring and deadlock-monitoring levels).
- The type of lock held on the object by this application (for object-lock levels).

The values for this field are:

Mode	Type of Lock	API Constant
	No Lock	SQLM_LNON
IS	Intention Share Lock	SQLM_LOIS
IX	Intention Exclusive Lock	SQLM_LOIX

Mode	Type of Lock	API Constant
S	Share Lock	SQLM_LOOS
SIX	Share with Intention Exclusive Lock	SQLM_LSIX
X	Exclusive Lock	SQLM_LOOX
IN	Intent None	SQLM_LOIN
Z	Super Exclusive Lock	SQLM_LOOZ
U	Update Lock	SQLM_LOOU
NS	Scan Share Lock	SQLM_LONS
NW	Next Key Weak Exclusive Lock	SQLM_LONW

## lock\_mode\_requested - Lock mode requested monitor element

The lock mode being requested by the application.

*Table 577. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Lock	lock_wait	Lock

*Table 578. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks <sup>1</sup>	event_dlconn	-
Deadlocks with Details <sup>1</sup>	event_detailed_dlconn	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

### Usage

The mode in which the lock was requested by the application. This value can help you determine the source of contention for resources.

## lock\_name - Lock name monitor element

Internal binary lock name. This element serves as a unique identifier for locks.

*Table 579. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Lock	lock	Basic
Lock	lock_wait	lock_wait

*Table 580. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-

Table 580. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks <sup>1</sup>	lock	-
Deadlocks <sup>1</sup>	event_dlconn	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

## lock\_node - Lock Node

The node involved in a lock.

Table 581. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement
Deadlocks	event_dlconn	Statement
Deadlocks with Details	event_detailed_dlconn	Statement

**Usage** This can be used for troubleshooting.

## lock\_object\_name - Lock Object Name

This element is provided for informational purposes only. It is the name of the object for which the application holds a lock (for object-lock-level information), or the name of the object for which the application is waiting to obtain a lock (for application-level and deadlock-level information).

**Note:** This monitor element has been deprecated. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

Table 582. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Lock
Lock	appl_lock_list	Lock
Lock	lock	Basic

Table 583. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	lock	-
Deadlocks	event_dlconn	-
Deadlocks with Details	event_detailed_dlconn	-

**Usage** For table-level locks, it is the file ID (FID) for SMS and DMS table spaces. For row-level locks, the object name is the row ID (RID). For table space locks, the object name is blank. For buffer pool locks, the object name is the name of the buffer pool.

To determine the table holding the lock, use *table\_name* and *table\_schema* instead of the file ID, since the file ID may not be unique.

To determine the table space holding the lock, use *tablespace\_name*.

---

## lock\_object\_type - Lock object type waited on monitor element

The type of object against which the application holds a lock (for object-lock-level information), or the type of object for which the application is waiting to obtain a lock (for application-level and deadlock-level information).

*Table 584. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Lock
Lock	appl_lock_list	Lock
Lock	lock	Basic
Lock	lock_wait	Lock

*Table 585. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks <sup>1</sup>	lock	-
Deadlocks <sup>1</sup>	event_dlconn	-
Deadlocks with Details <sup>1</sup>	event_detailed_dlconn	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

### Usage

This element can help you determine the source of contention for resources.

The object type identifiers are defined in *sqlmon.h*. The objects may be one of the following types:

- Table space (SQLM\_TABLESPACE\_LOCK in *sqlmon.h*)
- Table
- Buffer pool
- Block
- Record (or row)
- Data partition (SQLM\_TABLE\_PART\_LOCK in *sqlmon.h*)
- Internal (another type of lock held internally by the database manager)
- Automatic resize
- Automatic storage.

---

## lock\_release\_flags - Lock release flags monitor element

Lock release flags.

Table 586. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Lock	lock	Basic
Lock	lock_wait	Basic

Table 587. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks <sup>1</sup>	lock	-
Deadlocks <sup>1</sup>	event_dlconn	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

## Usage

The following are possible release flag settings. Each release flag is based upon a bit flag value defined in sqlmon.h.

API Constant	Description
SQLM_LOCKRELFIELDS_SQLCOMPILER	Locks by SQL compiler.
SQLM_LOCKRELFIELDS_UNTRACKED	Non-unique, untracked locks.

**Note:** All non-assigned bits are used for application cursors.

## lock\_status - Lock status monitor element

Indicates the internal status of the lock.

Table 588. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Lock	lock	Basic

Table 589. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks <sup>1</sup>	lock	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

## Usage

This element can help explain what is happening when an application is waiting to obtain a lock on an object. While it may appear that the application already has a lock on the object it needs, it may have to wait to obtain a different type of lock on the same object.

The lock can be in one of the following statuses:

### Granted state

The application has the lock in the state specified by the **lock\_mode** monitor element.

### Converting state

The application is trying to change the lock held to a different type; for example, changing from a share lock to an exclusive lock.

**Note:** API users should refer to the `sqlmon.h` header file containing definitions of database system monitor constants.

---

## lock\_timeout\_val - Lock timeout value monitor element

Indicates the timeout value (in seconds) when an application has issued a SET CURRENT LOCK TIMEOUT statement. In cases where the statement has not been executed, the database level lock timeout will be shown.

*Table 590. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic
Application	agent	Basic

*Table 591. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-

## Usage

The SET CURRENT LOCK TIMEOUT statement can be used to specify the maximum duration for which application agents will wait for a table or index lock.

If an application is waiting too long on a lock, you can check the **lock\_timeout\_val** monitor element value to see whether it is set too high inside the application. You can modify the application to lower the lock timeout value to let the application time out, if that is appropriate for the application logic. You can accomplish this modification with the SET CURRENT LOCK TIMEOUT statement.

If the application is timing out frequently, you can check whether the lock timeout value is set too low and increase it as appropriate.

---

## lock\_timeouts - Number of lock timeouts monitor element

The number of times that a request to lock an object timed out instead of being granted.

Table 592. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 593. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 594. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-



Table 594. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

## Usage

This element can help you adjust the setting for the **locktimeout** database configuration parameter. If the number of lock timeouts becomes excessive when compared to normal operating levels, you may have an application that is holding locks for long durations. In this case, this element may indicate that you should analyze some of the other lock and deadlock monitor elements to determine if you have an application problem.

You could also have too few lock timeouts if your **locktimeout** database configuration parameter is set too high. In this case, your applications may wait excessively to obtain a lock.

---

## lock\_wait\_start\_time - Lock Wait Start Timestamp

The date and time that this application started waiting to obtain a lock on the object that is currently locked by another application.

### Element identifier

lock\_wait\_start\_time

### Element type

timestamp

Table 595. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Lock, Timestamp
Lock	lock_wait	Lock, Timestamp

Table 596. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	event_dlconn	Timestamp
Deadlocks with Details	event_detailed_dlconn	Timestamp

**Usage** This element can help you determine the severity of resource contention.

---

## lock\_wait\_time - Time waited on locks monitor element

The total elapsed time spent waiting for locks. The value is given in milliseconds.

Table 597. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 597. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 598. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Lock
Application	appl	Lock
Lock	appl_lock_list	appl_lock_list

For snapshot monitoring, this counter can be reset.

Table 599. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Connection	event_conn	-
Transactions	event_xact	-

**Usage** At the database level, this is the total amount of elapsed time that all applications were waiting for a lock within this database.

At the application-connection and transaction levels, this is the total amount of elapsed time that this connection or transaction has waited for a lock to be granted to it.

The value for this element does not include lock wait times for agents that are currently still in a lock wait state. It only includes lock wait times for agents that have already completed their lock waits.

This element may be used in conjunction with the **lock\_waits** monitor element to calculate the average wait time for a lock. This calculation can be performed at either the database or the application-connection level.

When using monitor elements providing elapsed times, you should consider:

- Elapsed times are affected by system load, so the more processes you have running, the higher this elapsed time value.
- To calculate this element at the database level, the database system monitor sums the application-level times. This can result in double counting elapsed times at a database level, since more than one application process can be running at the same time.

To provide meaningful data, you can calculate the average wait time for a lock, as described above.

---

## lock\_wait\_time\_top – Lock wait time top monitor element

The high watermark for lock wait times of any request in a workload. Units are milliseconds. The `lock_wait_time_top` high watermark is always collected for workloads. A request contributes towards this high watermark only when request metrics are enabled.

*Table 600. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wlstats	-

### Usage

Use this element to determine the highest lock wait time of any request on a partition for a workload during the time interval collected.

---

## lock\_waits - Lock waits monitor element

The total number of times that applications or connections waited for locks.

*Table 601. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 601. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 602. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 603. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Connection	event_conn	-

**Usage** At the database level, this is the total number of times that applications have had to wait for locks within this database.

At the application-connection level, this is the total number of times that this connection requested a lock but had to wait because another connection was already holding a lock on the data.

This element may be used with **lock\_wait\_time** to calculate, at the database level, the average wait time for a lock. This calculation can be done at either the database or the application-connection level.

If the average lock wait time is high, you should look for applications that hold many locks, or have lock escalations, with a focus on tuning your applications to improve concurrency, if appropriate. If escalations are the reason for a high average lock wait time, then the values of one or both of the **locklist** and **maxlocks** configuration parameters may be too low.

---

## locks\_held - Locks Held

The number of locks currently held.

**Element identifier**  
locks\_held

**Element type**  
gauge

*Table 604. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic
Lock	db_lock_list	Basic
Lock	appl_lock_list	Basic

*Table 605. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks with Details	event_detailed_dlconn	-

**Usage** If the monitor information is at the database level, this is the total number of locks currently held by all applications in the database.

If the monitor information is at the application level, this is the total number of locks currently held by all agents for the application.

---

## locks\_held\_top - Maximum Number of Locks Held

The maximum number of locks held during this transaction.

**Element identifier**  
locks\_held\_top

**Element type**  
counter

*Table 606. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Transactions	event_xact	-

**Usage** You can use this element to determine if your application is approaching the maximum number of locks available to it, as defined by the *maxlocks* configuration parameter. This parameter indicates the percentage of the

lock list that each application can use before lock escalations occur. Lock escalations can result in a decrease in concurrency between applications connected to a database.

Since the *maxlocks* parameter is specified as a percentage and this element is a counter, you can compare the count provided by this element against the total number of locks that can be held by an application, as calculated using the following formula:

$$(\text{locklist} * 4096 / 36) * (\text{maxlocks} / 100)$$

If you have a large number of locks, you may need to perform more commits within your application so that some of the locks can be released.

## locks\_in\_list - Number of Locks Reported

The number of locks held by a particular application to be reported on by the event monitor.

Table 607. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks with Details	event_detailed_dlconn	-

## locks\_waiting - Current Agents Waiting On Locks

Indicates the number of agents waiting on a lock.

Table 608. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic
Lock	db_lock_list	Basic

**Usage** When used in conjunction with **appls\_cur\_cons**, this element indicates the percentage of applications waiting on locks. If this number is high, the applications may have concurrency problems, and you should identify applications that are holding locks or exclusive locks for long periods of time.

## log\_buffer\_wait\_time - Log buffer wait time monitor element

The amount of time an agent spends waiting for space in the log buffer. The value is given in milliseconds.

Table 609. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 609. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 610. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## log\_disk\_wait\_time - Log disk wait time monitor element

The amount of time an agent spends waiting for log records to be flushed to disk. The value is given in milliseconds.

Table 611. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 611. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 612. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## log\_disk\_waits\_total - Total log disk waits monitor element

The number of times agents have to wait for log data to write to disk.

Table 613. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE



Table 613. Table Function Monitoring Information (continued)

Table Function	Monitor Element	Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS	BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS	BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS	BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS	BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS	BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS	BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS	BASE

Table 614. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## log\_held\_by\_dirty\_pages - Amount of Log Space Accounted for by Dirty Pages

The amount of log (in bytes) corresponding to the difference between the oldest dirty page in the database and the top of the active log.

### Element identifier

log\_held\_by\_dirty\_pages

### Element type

watermark

Table 615. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Table 616. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

**Usage** When the snapshot is taken, this value is calculated based on conditions at the time of that snapshot.

Use this element to evaluate the effectiveness of page cleaning for older pages in the buffer pool.

The cleaning of old pages in the buffer pool is governed by the *softmax* database configuration parameter. If the page cleaning is effective then *log\_held\_by\_dirty\_pages* should be less than or approximately equal to:

$$(\text{softmax} / 100) * \text{logfilesiz} * 4096$$

If this statement is not true, increase the number of page cleaners (*num\_iocleaners*) configuration parameter.

If the condition is true and it is desired that less log be held by dirty pages, then decrease the *softmax* configuration parameter.

---

## log\_read\_time - Log Read Time

The total elapsed time spent by the logger reading log data from the disk.

Table 617. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

For snapshot monitoring, this counter can be reset.

Table 618. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

**Usage** Use this element in conjunction with the *log\_reads*, *num\_log\_read\_io*, and *num\_log\_data\_found\_in\_buffer* elements to determine if:

- The current disk is adequate for logging.
- The log buffer size is adequate.

---

## log\_reads - Number of Log Pages Read

The number of log pages read from disk by the logger.

**Element identifier**

log\_reads

**Element type**

counter

Table 619. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

For snapshot monitoring, this counter can be reset.

Table 620. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

**Usage** You can use this element with an operating system monitor to quantify the amount of I/O on a device that is attributable to database activity.

---

## log\_to\_redo\_for\_recovery - Amount of Log to be Redone for Recovery

The amount of log (in bytes) that will have to be redone for crash recovery.

**Element identifier**

log\_to\_redo\_for\_recovery

**Element type**

watermark

Table 621. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Table 622. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

**Usage** When the snapshot is taken, this value is calculated based on conditions at the time of that snapshot. Larger values indicate longer recovery times after a system crash. If the value seems excessive, check the *log\_held\_by\_dirty\_pages* monitor element to see if page cleaning needs to be tuned. Also check if there are any long running transactions that need to be terminated.

---

## log\_write\_time - Log Write Time

The total elapsed time spent by the logger writing log data to the disk.

Table 623. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

For snapshot monitoring, this counter can be reset.

Table 624. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

**Usage** Use this element in conjunction with the *log\_writes* and *num\_log\_write\_io* elements to determine if the current disk is adequate for logging.

---

## log\_writes - Number of Log Pages Written

The number of log pages written to disk by the logger.

**Element identifier**

log\_writes

**Element type**

counter

*Table 625. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

For snapshot monitoring, this counter can be reset.

*Table 626. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

**Usage** You may use this element with an operating system monitor to quantify the amount of I/O on a device that is attributable to database activity.

**Note:** When log pages are written to disk, the last page might not be full. In such cases, the partial log page remains in the log buffer, and additional log records are written to the page. Therefore log pages might be written to disk by the logger more than once. You should not use this element to measure the number of pages produced by DB2.

---

## long\_object\_pages - Long Object Pages

The number of disk pages consumed by long data in a table.

*Table 627. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table	Basic

*Table 628. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Tables	event_table	-

**Usage** This element provides a mechanism for viewing the actual amount of space consumed by long data in a particular table. This element can be used in conjunction with a table event monitor to track the rate of long data growth over time.

---

## long\_tbsp\_id - Long table space ID monitor element

An identifier of the table space that holds long data (LONG or LOB type columns) for this table.

*Table 629. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLE table function - Get table metrics	Always collected

## Usage

The value of this element matches a value from column TBSPACEID of view SYSCAT.TABLESPACES.

---

## max\_agent\_overflows - Maximum Agent Overflows

The number of times a request to create a new agent was received when the Maximum Number of Agents (**maxagents**) configuration parameter had already been reached.

**Note:** The **max\_agent\_overflows** monitor element is deprecated starting with DB2 Version 9.5. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

*Table 630. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

## Usage

If agent creation requests are still being received when the **maxagents** configuration parameter has been reached, this might indicate too high a workload for this node.

---

## max\_data\_received\_1024 - Number of Statements with Outbound Bytes Received Between 513 and 1024 Bytes

This element represents the number of statements or chains with outbound bytes received between 513 and 1024 inclusive.

### Element identifier

max\_data\_received\_1024

### Element type

counter

*Table 631. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## max\_data\_received\_128 - Number of Statements with Outbound Bytes Received Between 1 and 128 Bytes

This element represents the number of statements or chains with outbound bytes received between 1 and 128 inclusive.

**Element identifier**

max\_data\_received\_128

**Element type**

counter

*Table 632. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## max\_data\_received\_16384 - Number of Statements with Outbound Bytes Received Between 8193 and 16384 Bytes

This element represents the number of statements or chains with outbound bytes received between 8193 and 16384 inclusive.

**Element identifier**

max\_data\_received\_16384

**Element type**

counter

*Table 633. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## max\_data\_received\_2048 - Number of Statements with Outbound Bytes Received Between 1025 and 2048 Bytes

This element represents the number of statements or chains with outbound bytes received between 1025 and 2048 inclusive.

**Element identifier**

max\_data\_received\_2048

**Element type**  
counter

*Table 634. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dc_s_dbase	Statement
DCS Application	dc_s_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## **max\_data\_received\_256 - Number of Statements with Outbound Bytes Received Between 129 and 256 Bytes**

This element represents the number of statements or chains with outbound bytes received between 129 and 256 inclusive.

**Element identifier**  
max\_data\_received\_256

**Element type**  
counter

*Table 635. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dc_s_dbase	Statement
DCS Application	dc_s_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## **max\_data\_received\_31999 - Number of Statements with Outbound Bytes Received Between 16385 and 31999 Bytes monitor element**

This element represents the number of statements or chains with outbound bytes received between 16385 and 31999 inclusive.

*Table 636. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dc_s_dbase	Statement
DCS Application	dc_s_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## **max\_data\_received\_4096 - Number of Statements with Outbound Bytes Received Between 2049 and 4096 Bytes**

This element represents the number of statements or chains with outbound bytes received between 2049 and 4096 inclusive.

**Element identifier**

max\_data\_received\_4096

**Element type**

counter

*Table 637. Snapshot Monitoring Information*

<b>Snapshot Level</b>	<b>Logical Data Grouping</b>	<b>Monitor Switch</b>
DCS Database	dcс_dbase	Statement
DCS Application	dcс_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## **max\_data\_received\_512 - Number of Statements with Outbound Bytes Received Between 257 and 512 Bytes**

This element represents the number of statements or chains with outbound bytes received between 257 and 512 inclusive.

**Element identifier**

max\_data\_received\_512

**Element type**

counter

*Table 638. Snapshot Monitoring Information*

<b>Snapshot Level</b>	<b>Logical Data Grouping</b>	<b>Monitor Switch</b>
DCS Database	dcс_dbase	Statement
DCS Application	dcс_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## **max\_data\_received\_64000 - Number of Statements with Outbound Bytes Received Between 32000 and 64000 Bytes monitor element**

This element represents the number of statements or chains with outbound bytes received between 32000 and 64000 inclusive.



Table 639. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## max\_data\_received\_8192 - Number of Statements with Outbound Bytes Received Between 4097 and 8192 Bytes

This element represents the number of statements or chains with outbound bytes received between 4097 and 8192 inclusive.

**Element identifier**

max\_data\_received\_8192

**Element type**

counter

Table 640. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## max\_data\_received\_gt64000 - Number of Statements with Outbound Bytes Received Greater than 64000 Bytes

This element represents the number of statements or chains with outbound bytes received greater than 64000.

**Element identifier**

max\_data\_received\_gt64000

**Element type**

counter

Table 641. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## max\_data\_sent\_1024 - Number of Statements with Outbound Bytes Sent Between 513 and 1024 Bytes

This element represents the number of statements or chains with outbound bytes sent between 513 and 1024 inclusive.

**Element identifier**

max\_data\_sent\_1024

**Element type**

counter

*Table 642. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## max\_data\_sent\_128 - Number of Statements with Outbound Bytes Sent Between 1 and 128 Bytes

This element represents the number of statements or chains with outbound bytes sent between 1 and 128 inclusive.

**Element identifier**

max\_data\_sent\_128

**Element type**

counter

*Table 643. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## max\_data\_sent\_16384 - Number of Statements with Outbound Bytes Sent Between 8193 and 16384 Bytes

This element represents the number of statements or chains with outbound bytes sent between 8193 and 16384 inclusive.

**Element identifier**

max\_data\_sent\_16384

**Element type**

counter

*Table 644. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## max\_data\_sent\_2048 - Number of Statements with Outbound Bytes Sent Between 1025 and 2048 Bytes

This element represents the number of statements or chains with outbound bytes sent between 1025 and 2048 inclusive.

**Element identifier**

max\_data\_sent\_2048

**Element type**

counter

*Table 645. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## max\_data\_sent\_256 - Number of Statements with Outbound Bytes Sent Between 129 and 256 Bytes

This element represents the number of statements or chains with outbound bytes sent between 129 and 256 inclusive.

**Element identifier**

max\_data\_sent\_256

**Element type**  
counter

*Table 646. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcс_dbase	Statement
DCS Application	dcс_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## **max\_data\_sent\_31999 - Number of Statements with Outbound Bytes Sent Between 16385 and 31999 Bytes**

This element represents the number of statements or chains with outbound bytes sent between 16385 and 31999 inclusive.

**Element identifier**  
max\_data\_sent\_31999

**Element type**  
counter

*Table 647. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcс_dbase	Statement
DCS Application	dcс_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## **max\_data\_sent\_4096 - Number of Statements with Outbound Bytes Sent Between 2049 and 4096 Bytes**

This element represents the number of statements or chains with outbound bytes sent between 2049 and 4096 inclusive.

**Element identifier**  
max\_data\_sent\_4096

**Element type**  
counter

*Table 648. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcс_dbase	Statement
DCS Application	dcс_appl	Statement

Table 648. Snapshot Monitoring Information (continued)

Snapshot Level	Logical Data Grouping	Monitor Switch
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## max\_data\_sent\_512 - Number of Statements with Outbound Bytes Sent Between 257 and 512 Bytes

This element represents the number of statements or chains with outbound bytes sent between 257 and 512 inclusive.

**Element identifier**

max\_data\_sent\_512

**Element type**

counter

Table 649. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## max\_data\_sent\_64000 - Number of Statements with Outbound Bytes Sent Between 32000 and 64000 Bytes

This element represents the number of statements or chains with outbound bytes sent between 32000 and 64000 inclusive.

**Element identifier**

max\_data\_sent\_64000

**Element type**

counter

Table 650. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## max\_data\_sent\_8192 - Number of Statements with Outbound Bytes Sent Between 4097 and 8192 Bytes

This element represents the number of statements or chains with outbound bytes sent between 4097 and 8192 inclusive.

**Element identifier**

max\_data\_sent\_8192

**Element type**

counter

*Table 651. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## max\_data\_sent\_gt64000 - Number of Statements with Outbound Bytes Sent Greater than 64000 Bytes

This element represents the number of statements or chains with outbound bytes sent greater than 64000.

**Element identifier**

max\_data\_sent\_gt64000

**Element type**

counter

*Table 652. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## max\_network\_time\_100\_ms - Number of Statements with Network Time between 16 and 100 ms

This element represents the number of statements or chains whose network time was greater than 16 milliseconds but less or equal to 100 milliseconds. (Network time is the difference between host response time and elapsed execution time for a statement or chain.)

*Table 653. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcс_dbase	Statement
DCS Application	dcс_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## max\_network\_time\_16\_ms - Number of Statements with Network Time between 4 and 16 ms

This element represents the number of statements or chains whose network time was greater than 4 milliseconds but less or equal to 16 milliseconds. (Network time is the difference between host response time and elapsed execution time for a statement or chain.)

**Element identifier**

max\_network\_time\_16\_ms

**Element type**

counter

*Table 654. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcс_dbase	Statement
DCS Application	dcс_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## max\_network\_time\_1\_ms - Number of Statements with Network Time of up to 1 ms

This element represents the number of statements or chains whose network time was less or equal to 1 millisecond. (Network time is the difference between host response time and elapsed execution time for a statement or chain.)

Table 655. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcс_dbase	Statement
DCS Application	dcс_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## max\_network\_time\_4\_ms - Number of Statements with Network Time between 1 and 4 ms

This element represents the number of statements or chains whose network time was greater than 1 millisecond but less or equal to 4 milliseconds. (Network time is the difference between host response time and elapsed execution time for a statement or chain.)

**Element identifier**

max\_network\_time\_4\_ms

**Element type**

counter

Table 656. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcс_dbase	Statement
DCS Application	dcс_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## max\_network\_time\_500\_ms - Number of Statements with Network Time between 100 and 500 ms

This element represents the number of statements or chains whose network time was greater than 100 milliseconds but less or equal to 500 milliseconds. (Network time is the difference between host response time and elapsed execution time for a statement or chain.)

Table 657. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcс_dbase	Statement
DCS Application	dcс_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.



**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## max\_network\_time\_gt500\_ms - Number of Statements with Network Time greater than 500 ms

This element represents the number of statements or chains whose network time was greater than 500 milliseconds. (Network time is the difference between host response time and elapsed execution time for a statement or chain.)

*Table 658. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Statement
DCS Application	dcs_appl	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## member - Database member monitor element

The numeric identifier for the database member from which the data was retrieved for this result record.

*Table 659. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics	Always collected
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics	Always collected
MON_GET_CONNECTION table function - Get connection metrics	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_WORKLOAD table function - Get workload metrics	Always collected
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	Always collected
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_TABLE table function - Get table metrics	Always collected

Table 659. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics	Always collected
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details	Always collected
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics	Always collected
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected
MON_GET_EXTENT_MOVEMENT_STATUS - Get extent movement progress status metrics	Always collected
MON_GET_INDEX table function - Get index metrics	Always collected

Table 660. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

A DB2 member is a database manager instance that runs DB2 server software on a single host, A DB2 member accepts and processes database requests from applications connected to it.

## message - Control Table Message

The nature of the timestamp in the MESSAGE\_TIME column. This element is only used in the CONTROL table by write-to-table event monitors.

Table 661. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
-	-	-

## Usage

The following are possible values:

### **DROPPED RECORDS:** *n*

Number of activity records that were dropped because MONHEAP could not be allocated for them.

**FIRST\_CONNECT**

The time of the first connect to the database after activation.

**EVMON\_START**

The time the event monitor listed in the EVMONNAME column was started.

**OVERFLOWS: *n***

Denotes that *n* records were discarded due to buffer overflow.

**LAST DROPPED RECORD**

The last time that an activity record was dropped.

**message\_time - Timestamp Control Table Message**

The timestamp corresponding to the event described in the MESSAGE column. This element is only used in the CONTROL table by write-to-table event monitors.

*Table 662. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
-	-	-

**nesting\_level - Nesting level monitor element**

This represents the nesting level of this activity. Nesting level is the depth to which this activity is nested within its top-most parent activity.

*Table 663. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

**Usage****network\_time\_bottom - Minimum Network Time for Statement**

This element represents the shortest network time for a statement executed against this DCS database or in this DCS application, or having used this many data transmissions. (Network time is the difference between host response time and elapsed execution time for a statement.)

**Element identifier**

network\_time\_bottom

**Element type**

watermark

*Table 664. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcс_dbase	Statement, Timestamp
DCS Application	dcс_appl	Statement, Timestamp
Data Transmission	stmt_transmissions	Statement, Timestamp

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels.

---

## network\_time\_top - Maximum Network Time for Statement

This element represents the longest network time for a statement executed against this DCS database or in this DCS application, or having used this many data transmissions. (Network time is the difference between host response time and elapsed execution time for a statement.)

**Element identifier**

network\_time\_top

**Element type**

watermark

*Table 665. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dc_s_dbase	Statement, Timestamp
DCS Application	dc_s_appl	Statement, Timestamp
Data Transmission	stmt_transmissions	Statement, Timestamp

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to get a better idea of the database activity and network traffic at the database or application levels. Note that this element is not collected when the timestamp switch is off.

---

## nleaf - Number of leaf pages monitor element

The approximate number of leaf pages.

*Table 666. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

---

## nlevels - Number of index levels monitor element

Number of index levels. This is an approximation.

*Table 667. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

---

## node\_number - Node Number

The number assigned to the node in the *db2nodes.cfg* file.

Table 668. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	collected	Basic
Database Manager	memory_pool	Basic
Database Manager	fcm	Basic
Database Manager	fcm_node	Basic
Database Manager	utility_info	Basic
Database	detail_log	Basic
Buffer Pool	bufferpool_nodeinfo	Buffer Pool
Table Space	rollforward	Basic
Lock	lock	Basic
Lock	lock_wait	Basic
Database	db_sto_path_info	Buffer Pool

Table 669. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connections	event_connheader	-
Deadlocks	lock	-
Overflow Record	event_overflow	-
Database	event_dbmemuse	-
Connection	event_connmemuse	-

**Usage** This value identifies the current node number, which can be used when monitoring multiple nodes.

## nonboundary\_leaf\_node\_splits - Non-boundary leaf node splits monitor element

Number of times a non-boundary leaf node was split during an insert operation.

Table 670. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

## num\_agents - Number of Agents Working on a Statement

Number of concurrent agents currently executing a statement or subsection.

**Element identifier**  
num\_agents

**Element type**  
gauge

Table 671. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement

Table 671. Snapshot Monitoring Information (continued)

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

**Usage** An indicator how well the query is parallelized. This is useful for tracking the progress of query execution, by taking successive snapshots.

---

## num\_assoc\_agents - Number of Associated Agents

At the application level, this is the number of subagents associated with an application. At the database level, it is the number of subagents for all applications.

**Element identifier**

num\_assoc\_agents

**Element type**

gauge

Table 672. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl_info	Basic

**Usage** You can use this element to help evaluate your settings for your agent configuration parameters.

---

## num\_compilations - Statement Compilations

The number of different compilations for a specific SQL statement.

Table 673. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Dynamic SQL	dynsql	Basic

**Usage** Some SQL statements issued on different schemas, such as "select t1 from foo" will appear to be the same statement in the DB2 cache even though they refer to different access plans. Use this value in conjunction with num\_executions to determine whether a bad compilation environment may be skewing the results of dynamic SQL snapshot statistics.

---

## num\_db\_storage\_paths - Number of automatic storage paths

This element shows the number of automatic storage paths associated with this database.

Table 674. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

**Usage** You can use this element with the db\_storage\_path monitor element to identify the storage paths that are associated with this database.

---

## num\_executions - Statement executions monitor element

The number of times that an SQL statement has been executed.

*Table 675. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

*Table 676. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Dynamic SQL	dynsql	Basic

For snapshot monitoring, this counter can be reset.

### Usage

You can use this element to identify the most frequently executed SQL statements in your system.

At the package cache level, use this element to compute averages for the activity metrics reported per statement. For example, the average CPU usage for an execution of a statement reported at the package cache level can be calculated by the following formula:

`total_cpu_time / num_exec_with_metrics`

Use the **num\_exec\_with\_metrics** monitor element instead of the **num\_executions** monitor element when computing averages, since the **num\_executions** monitor element counts all executions of a statement, regardless of whether or not the execution of the statement contributed to the activity metrics that are reported.

---

## num\_exec\_with\_metrics - Number of executions with metrics collected monitor element

The number of times that this SQL statement section has been executed with the metrics collected. This element can be used to calculate the per execution value for monitor elements for statements in the package cache.

*Table 677. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

---

## num\_extents\_left - Number of extents left to process monitor element

The number of extents left to move during this table rebalancing process.

Table 678. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_EXTENT_MOVEMENT_STATUS - Get extent movement progress status metrics	Always collected

---

## num\_extents\_moved - Number of extents moved monitor element

The number of extents moved so far during this extent movement operation.

Table 679. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_EXTENT_MOVEMENT_STATUS - Get extent movement progress status metrics	Always collected

---

## num\_gw\_conn\_switches - Connection Switches

The number of times that an agent from the agents pool was primed with a connection and was reassigned for use with a different DRDA database.

Table 680. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

### Usage

For most users, the default setting of the **num\_poolagents** configuration parameter ensures optimal performance. The default setting for this configuration parameter automatically manages agent pooling and avoids reassigning agents.

To reduce the value of this monitor element, adjust the value of the **num\_poolagents** configuration parameter.

---

## num\_indoubt\_trans - Number of Indoubt Transactions

The number of outstanding indoubt transactions in the database.

Table 681. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

**Usage** Indoubt transactions hold log space for uncommitted transactions, which can cause the logs to become full. When the logs are full, further transactions cannot be completed. The resolution of this problem involves a manual process of heuristically resolving the indoubt transactions. This monitor element provides a count of the number of currently outstanding indoubt transactions that must be heuristically resolved.



## num\_log\_buffer\_full - Number of full log buffers monitor element

The number of times agents have to wait for log data to write to disk while copying log records into the log buffer. This value is incremented per agent per incident. For example, if two agents attempt to copy log data while the buffer is full, then this value is incremented by two.

*Table 682. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

*Table 683. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

For snapshot monitoring, this counter can be reset.

*Table 684. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE

Table 684. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Use this element to determine if the **logbufsz** database configuration parameter needs to be increased.

---

## num\_log\_data\_found\_in\_buffer - Number of Log Data Found In Buffer

The number of times an agent reads log data from the buffer. Reading log data from the buffer is preferable to reading from the disk because the latter is slower.

Table 685. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

For snapshot monitoring, this counter can be reset.

Table 686. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

**Usage** Use this element in conjunction with the *num\_log\_read\_io* element to determine if the LOGBUFSSZ database configuration parameter needs to be increased.

---

## num\_log\_part\_page\_io - Number of Partial Log Page Writes

The number of I/O requests issued by the logger for writing partial log data to the disk.

Table 687. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

For snapshot monitoring, this counter can be reset.

Table 688. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

**Usage** Use this element in conjunction with the *log\_writes*, *log\_write\_time*, and *num\_log\_write\_io* elements to determine if the current disk is adequate for logging.

---

## num\_log\_read\_io - Number of Log Reads

The number of I/O requests issued by the logger for reading log data from the disk.

*Table 689. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

For snapshot monitoring, this counter can be reset.

*Table 690. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

**Usage** Use this element in conjunction with the *log\_reads* and *log\_read\_time* elements to determine if the current disk is adequate for logging.

---

## num\_log\_write\_io - Number of Log Writes

The number of I/O requests issued by the logger for writing log data to the disk.

*Table 691. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

For snapshot monitoring, this counter can be reset.

*Table 692. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

**Usage** Use this element in conjunction with the *log\_writes* and *log\_write\_time* elements to determine if the current disk is adequate for logging.

---

## num\_nodes\_in\_db2\_instance - Number of Nodes in Partition

The number of nodes on the instance where the snapshot was taken.

*Table 693. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

*Table 694. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Event Log Header	event_log_header	-

**Usage** Use this element to determine the number of nodes for an instance. For non-partitioned system databases, this value will be 1.

---

## num\_remaps - Number of remaps monitor element

Count of the number of times this activity has been remapped. If num\_remaps is greater than zero, the service\_class\_id of this activity record is the ID of the last service class to which the activity was remapped.

Table 695. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
WLM_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 696. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

### Usage

Use this information to verify whether the activity was remapped the expected number of times.

---

## num\_threshold\_violations - Number of threshold violations monitor element

The number of threshold violations that have taken place in this database since it was last activated.

Table 697. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

For snapshot monitoring, this counter can be reset.

Table 698. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

### Usage

This element can be used to help determine whether or not thresholds are effective for this particular application or whether the threshold violations are excessive.

---

## num\_transmissions - Number of Transmissions

Number of data transmissions between the DB2 Connect gateway and the host that was used to process this DCS statement. (One data transmission consists of either one send or one receive.)

**Note:**

This is a legacy monitor element that is not relevant for DB2 UDB Version 8.1.2 or higher. If you are using DB2 UDB Version 8.1.2 or higher, refer to the **num\_transmissions\_group** monitor element.

**Element identifier**

num\_transmissions

**Element type**

counter

-->

*Table 699. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Statement	dcs_stmt	Statement

**Usage** Use this element to get a better understanding of the reasons why a particular statement took longer to execute. For example, a query returning a large result set may need many data transmissions to complete.

## num\_transmissions\_group - Number of Transmissions Group

The range of data transmissions between the DB2 Connect gateway and the host that was used to process this DCS statement. (One data transmission consists of either one send or one receive.)

*Table 700. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Statement	dcs_stmt	Statement

**Usage** Use this element to get a better understanding of the reasons why a particular statement took longer to execute. For example, a query returning a large result set may need many data transmissions to complete.

The constants representing the ranges of transmissions are described as follows and are defined in sqlmon.h.

API Constant	Description
SQLM_DCS_TRANS_GROUP_2	2 transmissions
SQLM_DCS_TRANS_GROUP_3TO7	3 to 7 transmissions
SQLM_DCS_TRANS_GROUP_8TO15	8 to 15 transmissions
SQLM_DCS_TRANS_GROUP_16TO64	16 to 64 transmissions
SQLM_DCS_TRANS_GROUP_GT64	Greater than 64 transmissions

## number\_in\_bin - Number in bin monitor element

This element holds the count of the number of activities or requests that fall within the histogram bin.

*Table 701. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_histogrambin	-

## Usage

Use this element to represent the height of a bin in the histogram.

---

## olap\_func\_overflows - OLAP Function Overflows monitor element

The number of times that OLAP function data exceeded the available sort heap space.

Table 702. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 703. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

## Usage

At the database level, use this element in conjunction with `total_olap_funcs` to calculate the percentage of OLAP functions that overflowed to disk. If this percentage is high and the performance of applications using OLAP functions needs to be improved, then you should consider increasing the sort heap size.

At the application level, use this element to evaluate OLAP function performance for individual applications.

---

## open\_cursors - Number of Open Cursors

The number of cursors currently open for an application.

Table 704. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dc_s_appl	Statement

**Usage** Use this element to assess how much memory is being allocated. The amount of memory allocated by the DB2 client, DB2 Connect, or the database agent on the target database is related to the number of cursors that are currently open. Knowing this information can help with capacity planning. For example, each open cursor that is doing blocking has a buffer size of `RQRI0BLK`. If `deferred_prepare` is enabled, then two buffers will be allocated.

This element does not include cursors that were closed by an early close. An early close occurs when the host database returns the last record to the client. The cursor is closed at the host and gateway, but is still open at the client. Early close cursors can be set using the DB2 Call Level Interface.

---

## open\_loc\_curs - Open Local Cursors

The number of local cursors currently open for this application, including those cursors counted by *open\_loc\_curs\_blk*.

**Element identifier**

open\_loc\_curs

**Element type**

gauge

Table 705. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

**Usage** You may use this element in conjunction with *open\_loc\_curs\_blk* to calculate the percentage of local cursors that are blocking cursors. If the percentage is low, you may be able to improve performance by improving the row blocking in the application.

For cursors used by remote applications, see *open\_rem\_curs*.

---

## open\_loc\_curs\_blk - Open Local Cursors with Blocking

The number of local blocking cursors currently open for this application.

**Element identifier**

open\_loc\_curs\_blk

**Element type**

gauge

Table 706. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

**Usage** You may use this element in conjunction with *open\_loc\_curs* to calculate the percentage of local cursors that are blocking cursors. If the percentage is low, you may be able to improve performance by improving the row blocking in the application:

- Check the pre-compile options for record blocking for treatment of ambiguous cursors
- Redefine cursors to allow for blocking (for example, if possible, specify FOR FETCH ONLY on your cursors).

*rej\_curs\_blk* and *acc\_curs\_blk* provide additional information that may help you tune your configuration parameters to improve row blocking in your application.

For blocking cursors used by remote applications, see *open\_rem\_curs\_blk*.

---

## open\_rem\_curs - Open Remote Cursors

The number of remote cursors currently open for this application, including those cursors counted by *open\_rem\_curs\_blk*.

**Element identifier**

open\_rem\_curs

**Element type**  
gauge

Table 707. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

**Usage** You may use this element in conjunction with *open\_rem\_curs\_blk* to calculate the percentage of remote cursors that are blocking cursors. If the percentage is low, you may be able to improve performance by improving the row blocking in the application. See *open\_rem\_curs\_blk* for more information.

For the number of open cursors used by applications connected to a local database, see *open\_loc\_curs*.

---

## open\_rem\_curs\_blk - Open Remote Cursors with Blocking

The number of remote blocking cursors currently open for this application.

**Element identifier**  
open\_rem\_curs\_blk

**Element type**  
gauge

Table 708. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

**Usage** You can use this element in conjunction with *open\_rem\_curs* to calculate the percentage of remote cursors that are blocking cursors. If the percentage is low, you may be able to improve performance by improving the row blocking in the application:

- Check the pre-compile options for record blocking for treatment of ambiguous cursors
- Redefine cursors to allow for blocking (for example, if possible, specify FOR FETCH ONLY on your cursors).

*rej\_curs\_blk* and *acc\_curs\_blk* provide additional information that may help you tune your configuration parameters to improve row blocking in your application.

For the number of open blocking cursors used by applications connected to a local database see *open\_loc\_curs\_blk*.

---

## outbound\_appl\_id - Outbound Application ID

This identifier is generated when the application connects to the DRDA host database. It is used to connect the DB2 Connect gateway to the host, while the **appl\_id** monitor element is used to connect a client to the DB2 Connect gateway.

**Note:** NetBIOS is no longer supported. SNA, including its APIs APPC, APPN, and CPI-C, is also no longer supported. If you use these protocols, you must recatalog your nodes and databases using a supported protocol such as TCP/IP. References to these protocols should be ignored.



**Element identifier**  
outbound\_appl\_id

**Element type**  
information

-->

*Table 709. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcs_appl_info	Basic

## Usage

You may use this element in conjunction with **appl\_id** to correlate the client and server parts of the application information.

This identifier is unique across the network.

This element will be blank when the gateway concentrator is on, or if the DCS application is not in a logical unit of work.

### Format

Network.LU Name.Application instance

### Example

CAIBMTOR.OSFDBM0.930131194520

### Details

This application ID is the displayable format of an actual SNA LUWID (Logical Unit-of-Work ID) that flows on the network when an APPC conversation is allocated. APPC-generated application IDs are made up by concatenating the network name, the LU name, and the LUWID instance number, which creates a unique label for the client/server application. The network name and LU name can each be a maximum of 8 characters. The application instance corresponds to the 12-decimal-character LUWID instance number.

---

## outbound\_bytes\_received - Outbound Number of Bytes Received

The number of bytes received by the DB2 Connect gateway from the host, excluding communication protocol overhead (for example, TCP/IP or SNA headers). For the data transmission level: Number of bytes received by the DB2 Connect gateway from the host during the processing of all the statements that used this number of data transmissions.

*Table 710. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Basic
DCS Application	dcs_appl	Basic
DCS Statement	dcs_stmt	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring at the statement level, this counter cannot be reset. This counter can be reset at other levels.

## Usage

Use this element to measure the throughput from the host databases to the DB2 Connect gateway.

---

### outbound\_bytes\_received\_bottom - Minimum Outbound Number of Bytes Received

The lowest number of bytes received per statement or chain by the DB2 Connect gateway from the host during the processing of all the statements or chains against this DCS database, or in this DCS application, that used this number of data transmissions.

Table 711. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Data Transmission	stmt_transmissions	Statement

**Usage** Use this element in conjunction with "outbound number of bytes received" as yet another parameter that illustrates the throughput from the host database to the DB2 Connect gateway.

---

### outbound\_bytes\_received\_top - Maximum Outbound Number of Bytes Received

Maximum number of bytes received per statement or chain by the DB2 Connect gateway from the host during the processing of all the statements or chains against this DCS database, or in this DCS application, that used this number of data transmissions.

Table 712. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Data Transmission	stmt_transmissions	Statement

**Usage** Use this element in conjunction with "outbound number of bytes received" as yet another parameter that illustrates the throughput from the host database to the DB2 Connect gateway.

---

### outbound\_bytes\_sent - Outbound Number of Bytes Sent

The number of bytes sent by the DB2 Connect gateway to the host, excluding communication protocol overhead (for example, TCP/IP or SNA headers). For the data transmission level: Number of bytes sent by the DB2 Connect gateway to the host during the processing of all the statements that used this number of data transmissions.

Table 713. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Basic
DCS Application	dcs_appl	Basic
DCS Statement	dcs_stmt	Statement
Data Transmission	stmt_transmissions	Statement

For snapshot monitoring at the statement level, this counter cannot be reset. This counter can be reset at other levels.

**Usage** Use this element to measure the throughput from the DB2 Connect gateway to the host database.

---

## outbound\_bytes\_sent\_bottom - Minimum Outbound Number of Bytes Sent

The lowest number of bytes sent per statement or chain by the DB2 Connect gateway to the host during the processing of all the statements or chains against this DCS database, or in this DCS application, that used this number of data transmissions.

*Table 714. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Data Transmission	stmt_transmissions	Statement

**Usage** Use this element in conjunction with "outbound number of bytes sent" as yet another parameter that illustrates the throughput from the DB2 Connect Gateway to the host database.

---

## outbound\_bytes\_sent\_top - Maximum Outbound Number of Bytes Sent

Maximum number of bytes sent per statement or chain by the DB2 Connect gateway to the host during the processing of all the statements or chains against this DCS database, or in this DCS application, that used this number of data transmissions.

*Table 715. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Data Transmission	stmt_transmissions	Statement

**Usage** Use this element in conjunction with "outbound number of bytes sent" as yet another parameter that illustrates the throughput from the DB2 Connect Gateway to the host database.

---

## outbound\_comm\_address - Outbound Communication Address

This is the communication address of the target database. For example, it could be an SNA net ID and LU partner name, or an IP address and port number for TCP/IP.

**Element identifier**

outbound\_comm\_address

**Element type**

information

*Table 716. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
DCS Application	dcs_appl_info	Basic

**Usage** Use this element for problem determination on DCS applications.

---

## outbound\_comm\_protocol - Outbound Communication Protocol

The communication protocol used between the DB2 Connect gateway and the host.

Table 717. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcs_appl_info	Basic

### Usage

Use this element for problem determination on DCS applications. The valid value is:

- SQLM\_PROT\_TCPIP

---

## outbound\_sequence\_no - Outbound Sequence Number

This element will be blank when the gateway concentrator is on, or if the DCS application is not in a logical unit of work.

Table 718. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcs_appl_info	Basic

---

## overflow\_accesses - Accesses to overflowed records monitor element

The number of accesses (reads and writes) to overflowed rows of this table.

Table 719. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLE table function - Get table metrics	Always collected

Table 720. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table	Basic

For snapshot monitoring, this counter can be reset.

Table 721. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Tables	event_table	-

### Usage

Overflowed rows indicate that data fragmentation has occurred. If this number is high, you may be able to improve table performance by reorganizing the table using the REORG utility, which cleans up this fragmentation.

A row overflows if it is updated and no longer fits in the data page where it was originally written. This usually happens as a result of an update of a VARCHAR or an ALTER TABLE statement.

---

## overflow\_creates - Overflow creates monitor element

The number of overflowed rows created on this table.

Table 722. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLE table function - Get table metrics	Always collected

### Usage

---

## package\_name - Package name monitor element

The name of the package that contains the SQL statement currently executing.

Table 723. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected

Table 724. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement
DCS Statement	dcs_stmt	Statement

Table 725. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details <sup>1</sup>	event_detailed_dlconn	-
Statements	event_stmt	-
Activities	event_activitystmt	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

### Usage

You may use this element to help identify the application program and the SQL statement that is executing.

---

## package\_schema - Package schema monitor element

If the activity is an SQL statement, this represents the schema name of its package.

Table 726. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected

Table 727. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-

---

## package\_version\_id - Package version monitor element

For a given package name and creator, there can exist (starting in DB2 Version 8) multiple versions. The package version identifies the version identifier of the package that contains the SQL statement currently executing. The version of a package is determined at precompile (PREP) of the embedded SQL program using the VERSION keyword. If not specified at precompile time the package version has a value of "" (empty string).

Table 728. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected

Table 729. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement

Table 730. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Statements	event_stmt	-
Activities	event_activitystmt	-

### Usage

Use this element to help identify the package and the SQL statement that is currently executing.

---

## page\_allocations - Page allocations monitor element

Number of pages that have been allocated to the index.

Table 731. Table Function Monitoring Information

Table Function	Monitor Element	Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected	

---

## page\_reorgs - Page Reorganizations

The number of page reorganizations executed for a table.

Table 732. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table	Basic

For snapshot monitoring, this counter can be reset.

Table 733. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Tables	event_table	-

### Usage

Although a page might have enough space, the page could become fragmented in the following situations:

- When a new row is inserted
- When an existing row is updated, and the update results in an increased record size

A page might require reorganization when it becomes fragmented. Reorganization moves all fragmented space to a contiguous area, where the new record can be written. Such a page reorganization (page reorg) might require thousands of instructions. It also generates a log record of the operation.

Too many page reorganizations can result in less than optimal insert performance. You can use the REORG TABLE utility to reorganize a table and eliminate fragmentation. You can also use the APPEND parameter for the ALTER TABLE statement to indicate that all inserts are appended at the end of a table to avoid page reorgs.

In situations where updates to rows causes the row length to increase, the page may have enough space to accommodate the new row, but a page reorg may be required to defragment that space. If the page does not have enough space for the new larger row, an overflow record is created causing *overflow\_accesses* during reads. You can avoid both situations by using fixed length columns instead of varying length columns.

---

## pages\_from\_block\_ios - Total number of pages read by block I/O monitor element

The total number of pages read by block I/O into the block area of the buffer pool.

Table 734. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 735. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool	Buffer Pool

## Usage

If block-based buffer pool is enabled, this element reports the total number of pages read by block I/O. Otherwise, this element returns 0.

To calculate the average number of pages sequentially prefetched per block-based I/O, divide the value of the **pages\_from\_block\_ios** monitor element by the value of the **block\_ios** monitor element. If this value is much less than the BLOCKSIZE option you have defined for the block-based buffer pool in the CREATE BUFFERPOOL or ALTER BUFFERPOOL statement, then block-based I/O is not being used to its full advantage. One possible cause for this is a mismatch between the extent size for the table space being sequentially prefetched and the block size of the block-based buffer pool.

---

## pages\_from\_vectored\_ios - Total number of pages read by vectored I/O monitor element

The total number of pages read by vectored I/O into the page area of the buffer pool.

Table 736. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 737. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool	Buffer Pool



---

## pages\_merged - Pages merged monitor element

Number of index pages that have been merged.

*Table 738. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

---

---

## pages\_read - Number of pages read monitor element

The number of pages (data, index, and XML) read in from the physical table space containers for regular and large table spaces.

*Table 739. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE

---

### Usage

---

## pages\_written - Number of pages written monitor element

The number of pages (data, index, and XML) physically written to the table space container.

*Table 740. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE

---

### Usage

---

## parent\_activity\_id - Parent activity ID monitor element

The unique ID of the activity's parent activity within the parent activity's unit of work. If there is no parent activity, the value of this monitor element is 0.

*Table 741. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

---

*Table 742. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

---

## Usage

Use this element along with the **parent\_uow\_id** element and **appl\_id** element to uniquely identify the parent activity of the activity described in this activity record.

---

### parent\_uow\_id - Parent unit of work ID monitor element

The unique unit of work identifier within an application handle. The ID of the unit of work in which the activity's parent activity originates. If there is no parent activity, the value is 0.

Table 743. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 744. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

## Usage

Use this element along with the **parent\_activity\_id** element and **appl\_id** element to uniquely identify the parent activity of the activity described in this activity record.

---

### partial\_record - Partial Record monitor element

Indicates that an event monitor record is only a partial record.

Table 745. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tables	event_table	-
Tablespaces	event_tablespace	-
Bufferpools	event_bufferpool	-
Connection	event_conn	-
Statements	event_stmt	-
Statements	event_subsection	-
Transactions	event_xact	-
Activities	event_activity	-

## Usage

Most event monitors do not output their results until database deactivation. You can use the `FLUSH EVENT MONITOR <monitorName>` statement to force monitor values to the event monitor output writer. This allows you to force event monitor

records to the writer without needing to stop and restart the event monitor. This element indicates whether an event monitor record was the result of flush operation and so is a partial record.

Flushing an event monitor does not cause its values to be reset. This means that a complete event monitor record is still generated when the event monitor is triggered.

At the event\_activity logical data grouping, the possible values of **partial\_record** monitor element are:

- 0 The activity record was generated normally at the end of activity.
- 1 The activity record was generated as a result of calling the WLM\_CAPTURE\_ACTIVITY\_IN\_PROGRESS stored procedure.
- 2 Information is missing for this activity because not enough storage was available to create the records. Information may be missing from the event\_activity, event\_activitystmt, or event\_activityvals records.

## participant\_no - Participant within Deadlock

A sequence number uniquely identifying this participant within this deadlock.

**Element identifier**

participant\_no

**Element type**

information

*Table 746. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	event_dlconn	-
Deadlocks with Details	event_detailed_dlconn	-

**Usage** Use this in your monitoring application to correlate deadlock connection event records with deadlock event records.

## participant\_no\_holding\_lk - Participant Holding a Lock on the Object Required by Application

The participant number of the application that is holding a lock on the object that this application is waiting to obtain.

**Element identifier**

participant\_no\_holding\_lk

**Element type**

information

*Table 747. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	event_dlconn	-
Deadlocks with Details	event_detailed_dlconn	-

**Usage** This element can help you determine which applications are in contention for resources.

---

## partition\_number - Partition Number

This element is only used in the target SQL tables by write-to-table event monitors in a partitioned database environment. This value indicates the number of the partition where event monitor data is inserted.

Table 748. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
-	-	-

---

## passthru\_time - Pass-Through Time

This element contains the aggregate amount of time, in milliseconds, that it has taken this data source to respond to PASSTHRU statements from all applications or a single application running on this federated server instance since the start of the federated server instance or the last reset of the database monitor counters, whichever is latest. The response time is measured as the difference between the time the federated server submits a PASSTHRU statement to the data source, and the time it takes the data source to respond, indicating that the statement has been processed.

Table 749. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Timestamp
Application	appl_remote	Timestamp

For snapshot monitoring, this counter can be reset.

### Usage

Use this element to determine how much actual time is spent at this data source processing statements in pass-through mode.

---

## passthru\_s - Pass-Through

This element contains a count of the total number of SQL statements that the federated server has passed through directly to this data source on behalf of any application since the start of the federated server instance or the last reset of the database monitor counters, whichever is latest.

Table 750. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to determine what percentage of your SQL statements can be handled natively by the federated server, and what percentage requires pass-through mode. If this value is high, you should determine the cause and investigate ways to better utilize native support.

---

## pipedsortsaccepted - Piped Sorts Accepted

The number of piped sorts that have been accepted.

Table 751. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

For snapshot monitoring, this counter can be reset.

**Usage** Each active sort on the system allocates memory, which may result in sorting taking up too much of the available system memory.

When the number of accepted piped sorts is low compared to the number requested, you can improve sort performance by adjusting one or both of the following configuration parameters:

- `sortheap`
- `sheapthres`

If piped sorts are being rejected, you might consider decreasing your sort heap or increasing your sort heap threshold. You should be aware of the possible implications of either of these options. If you increase the sort heap threshold, then there is the possibility that more memory will remain allocated for sorting. This could cause the paging of memory to disk. If you decrease the sort heap, you might require an extra merge phase that could slow down the sort.

---

## pipedsortsrequested - Piped Sorts Requested

The number of piped sorts that have been requested.

Table 752. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

For snapshot monitoring, this counter can be reset.

**Usage** Each active sort on the system allocates memory, which may result in sorting taking up too much of the available system memory.

The sort list heap (`sortheap`) and sort heap threshold (`sheapthres`) configuration parameters help to control the amount of memory used for sort operations. These parameters are also used to determine whether a sort will be piped.

Since piped sorts may reduce disk I/O, allowing more piped sorts can improve the performance of sort operations and possibly the performance of the overall system. A piped sort is not accepted if the sort heap threshold will be exceeded when the sort heap is allocated for the sort. See *pipedsortsaccepted* for more information if you are experiencing piped sort rejections.

The SQL EXPLAIN output will show whether the optimizer requests a piped sort.

---

## pkg\_cache\_inserts - Package Cache Inserts

The total number of times that a requested section was not available for use and had to be loaded into the package cache. This count includes any implicit prepares performed by the system.

**Element identifier**

pkg\_cache\_inserts

**Element type**

counter

*Table 753. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

*Table 754. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** In conjunction with "Package Cache Lookups", you can calculate the package cache hit ratio using the following formula:

$$1 - (\text{Package Cache Inserts} / \text{Package Cache Lookups})$$

See pkg\_cache\_lookups for information on using this element.

---

## pkg\_cache\_lookups - Package Cache Lookups

The number of times that an application looked for a section or package in the package cache. At a database level, it indicates the overall number of references since the database was started, or monitor data was reset. This counter includes the cases where the section is already loaded in the cache and when the section has to be loaded into the cache. In a concentrator environment where agents are being associated with different applications, additional package cache lookups may be required as a result of a new agent not having the required section or package available in local storage.

*Table 755. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

*Table 756. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Table 756. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

## Usage

To calculate the package cache miss ratio use the following formula:

$$1 - (\text{Package Cache Inserts} / \text{Package Cache Lookups})$$

The package cache miss ratio tells you whether or not the package cache is being used effectively. If the miss ratio is low (less than 0.2), the cache is performing well. A higher miss ratio may indicate that the package cache should be increased.

You will need to experiment with the size of the package cache to find the optimal number for the *pkcachesz* configuration parameter. For example, you might be able to use a smaller package cache size if there is no increase in the *pkg\_cache\_inserts* element when you decrease the size of the cache. Decreasing the package cache size frees up system resources for other work. It is also possible that you could improve overall system performance by increasing the size of the package cache if by doing so, you decrease the number of *pkg\_cache\_inserts*. This experimentation is best done under full workload conditions.

You can use this element with *ddl\_sql\_stmts* to determine whether or not the execution of DDL statements is impacting the performance of the package cache. Sections for dynamic SQL statements can become invalid when DDL statements are executed. Invalid sections are implicitly prepared by the system when next used. The execution of a DDL statement could invalidate a number of sections and the resulting extra overhead incurred when preparing those sections could significantly impact performance. In this case, the package cache hit ratio reflects the implicit recompilation of invalid sections. It does not reflect the insertion of new sections into the cache, so increasing the size of the package cache will not improve overall performance. You might find it less confusing to tune the cache for an application on its own before working in the full environment.

It is necessary to determine the role that DDL statements are playing in the value of the package cache hit ratio before deciding on what action to take. If DDL statements rarely occur, then cache performance may be improved by increasing its size. If DDL statements are frequent, then improvements may require that you limit the use of DDL statements (possibly to specific time periods).

The *static\_sql\_stmts* and *dynamic\_sql\_stmts* counts can be used to help provide information on the quantity and type of sections being cached.

See the *Administration Guide* for more information on the Package Cache Size (*pkcachesz*) configuration parameter.

**Note:** You may want to use this information at the database level to calculate the average package cache hit ratio all each applications. You should look at this information at an application level to find out the exact package cache hit ratio for a given application. It may not be worthwhile to increase the size of the package cache in order to satisfy the cache requirements of an application that only executes infrequently.

---

## pkg\_cache\_num\_overflows - Package Cache Overflows

The number of times that the package cache overflowed the bounds of its allocated memory.

Table 757. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

For snapshot monitoring, this counter can be reset.

Table 758. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

### Usage

Use this element with the **pkg\_cache\_size\_top** monitor element to determine whether the size of the package cache needs to be increased to avoid overflowing.

---

## pkg\_cache\_size\_top - Package cache high watermark

The largest size reached by the package cache.

**Note:** The **pkg\_cache\_size\_top** monitor element is deprecated starting with DB2 Version 9.5. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

Table 759. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Table 760. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

### Usage

If the package cache overflowed, then this element contains the largest size reached by the package cache during the overflow.

Check the **pkg\_cache\_num\_overflows** monitor element to determine if such a condition occurred.

You can determine the minimum size of the package cache required by your workload by:

```
maximum package cache size / 4096
```

Rounding the result up to a whole number, indicates the minimum number of 4K pages required by the package cache to avoid overflow.



## pool\_async\_data\_read\_reqs - Buffer pool asynchronous read requests monitor element

The number of asynchronous read requests by the prefetcher to the operating system. These requests are typically large block I/Os of multiple pages.

Table 761. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 762. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 763. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-

### Usage

To calculate the average number of data pages in each read request, use the following formula:

$$\text{pool\_async\_data\_reads} / \text{pool\_async\_data\_read\_reqs}$$

This average can help you determine the average read I/O size used by the prefetcher. This data can also be helpful in understanding the large block I/O requirements of the measured workload.

The maximum size of a prefetcher read I/O is the value specified on the EXTENTSIZE option of the CREATE TABLESPACE statement for the table space involved, but it can be smaller under some circumstances:

- when some pages of the extent are already in the buffer pool
- when exceeding operating system capabilities
- when the EXTENTSIZE option value is very large, such that doing a large I/O would be detrimental to overall performance

---

## pool\_async\_data\_reads - Buffer pool asynchronous data reads monitor element

Indicates the number of data pages read in from the table space containers (physical) by asynchronous engine dispatchable units (EDUs) for all types of table spaces.

Table 764. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 765. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 766. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-

### Usage

You can use this element with **pool\_data\_p\_reads** to calculate the number of physical reads that were performed synchronously (that is, physical data page reads that were performed by database manager agents). Use the following formula:

$$\frac{1 - ((\text{pool\_data\_p\_reads} + \text{pool\_index\_p\_reads}) - (\text{pool\_async\_data\_reads} + \text{pool\_async\_index\_reads}))}{(\text{pool\_data\_l\_reads} + \text{pool\_index\_l\_reads})}$$

By comparing the ratio of asynchronous to synchronous reads, you can gain insight into how well the prefetchers are working. This element can be helpful when you are tuning the **num\_ioservers** configuration parameter.

Asynchronous reads are performed by database manager prefetchers.

---

## pool\_async\_data\_writes - Buffer pool asynchronous data writes monitor element

The number of times a buffer pool data page was physically written to disk by either an asynchronous page cleaner, or a prefetcher. A prefetcher may have written dirty pages to disk to make space for the pages being prefetched.

Table 767. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 768. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 769. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-

**Usage** You can use this element with the **buffer\_pool\_data\_writes** monitor element to calculate the number of physical write requests that were performed synchronously (that is, physical data page writes that were performed by database manager agents). Use the following formula:

$$\text{pool\_data\_writes} - \text{pool\_async\_data\_writes}$$

By comparing the ratio of asynchronous to synchronous writes, you can gain insight into how well the buffer pool page cleaners are performing. This ratio can be helpful when you are tuning the **num\_iocleaners** configuration parameter.

## pool\_async\_index\_read\_reqs - Buffer pool asynchronous index read requests monitor element

The number of asynchronous read requests for index pages.

Table 770. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 771. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool

Table 771. Snapshot Monitoring Information (continued)

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 772. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-

**Usage** To calculate the number of index pages read per asynchronous request, use the following formula:

$$\text{pool\_async\_index\_reads} / \text{pool\_async\_index\_read\_reqs}$$

This average can help you determine the amount of asynchronous I/O done for index pages in each interaction with the prefetcher.

---

## pool\_async\_index\_reads - Buffer pool asynchronous index reads monitor element

Indicates the number of index pages read in from the table space containers (physical) by asynchronous engine dispatchable units (EDUs) for all types of table spaces.

Table 773. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 774. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 775. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-

## Usage

You can use this element with the **pool\_index\_p\_reads** monitor element to calculate the number of physical reads that were performed synchronously (that is, physical index page reads that were performed by database manager agents). Use the following formula:

$$1 - ((\text{pool\_data\_p\_reads} + \text{pool\_index\_p\_reads}) - (\text{pool\_async\_data\_reads} + \text{pool\_async\_index\_reads})) / (\text{pool\_data\_l\_reads} + \text{pool\_index\_l\_reads})$$

By comparing the ratio of asynchronous to synchronous reads, you can gain insight into how well the prefetchers are working. This element can be helpful when you are tuning the **num\_ioservers** configuration parameter.

Asynchronous reads are performed by database manager prefetchers.

---

## pool\_async\_index\_writes - Buffer pool asynchronous index writes monitor element

The number of times a buffer pool index page was physically written to disk by either an asynchronous page cleaner, or a prefetcher. A prefetcher may have written dirty pages to disk to make space for the pages being prefetched.

Table 776. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 777. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 778. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-

## Usage

You can use this element with the **pool\_index\_writes** monitor element to calculate the number of physical index write requests that were performed synchronously (that is, physical index page writes that were performed by database manager agents). Use the following formula:

$$\text{pool\_index\_writes} - \text{pool\_async\_index\_writes}$$

By comparing the ratio of asynchronous to synchronous writes, you can gain insight into how well the buffer pool page cleaners are performing. This ratio can be helpful when you are tuning the `num_iocleaners` configuration parameter.

---

## pool\_async\_read\_time - Buffer Pool Asynchronous Read Time

Indicates the total amount of time spent reading in data and index pages from the table space containers (physical) by asynchronous engine dispatchable units (EDUs) for all types of table spaces. This value is given in milliseconds.

**Element identifier**

pool\_async\_read\_time

**Element type**

counter

*Table 779. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

*Table 780. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-

**Usage** You can use this element to calculate the elapsed time for synchronous reading, using the following formula:

$$\text{pool\_read\_time} - \text{pool\_async\_read\_time}$$

You can also use this element to calculate the average asynchronous read time using the following formula:

$$\text{pool\_async\_read\_time} / \text{pool\_async\_data\_reads}$$

These calculations can be used to understand the I/O work being performed.

---

## pool\_async\_write\_time - Buffer Pool Asynchronous Write Time

The total elapsed time spent writing data or index pages from the buffer pool to disk by database manager page cleaners.

**Element identifier**

pool\_async\_write\_time

**Element type**

counter

*Table 781. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool

Table 781. Snapshot Monitoring Information (continued)

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 782. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespaces	-

**Usage** To calculate the elapsed time spent writing pages synchronously, use the following formula:

$$\text{pool\_write\_time} - \text{pool\_async\_write\_time}$$

You can also use this element to calculate the average asynchronous read time using the following formula:

$$\frac{\text{pool\_async\_write\_time}}{(\text{pool\_async\_data\_writes} + \text{pool\_async\_index\_writes})}$$

These calculations can be used to understand the I/O work being performed.

## pool\_async\_xda\_read\_reqs - Buffer pool asynchronous XDA read requests monitor element

The number of asynchronous read requests for XML storage object (XDA) data.

Table 783. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 784. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 785. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

Table 785. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Tablespaces	event_tablespace	-

**Usage** To calculate the average number of XML storage object data pages read per asynchronous request, use the following formula:

$$\text{pool\_async\_xda\_reads} / \text{pool\_async\_xda\_read\_reqs}$$

This average can help you determine the amount of asynchronous I/O done in each interaction with the prefetcher.

## pool\_async\_xda\_reads - Buffer pool asynchronous XDA data reads monitor element

Indicates the number of XML storage object (XDA) data pages read in from the table space containers (physical) by asynchronous engine dispatchable units (EDUs) for all types of table spaces.

Table 786. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 787. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 788. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-

### Usage

Use the **pool\_async\_xda\_reads** and **pool\_xda\_p\_reads** monitor elements to calculate the number of physical reads that were performed synchronously on XML storage object data pages (that is, physical data page reads that were performed by database manager agents on XML data). Use the following formula:

$$\text{pool\_xda\_p\_reads} - \text{pool\_async\_xda\_reads}$$



By comparing the ratio of asynchronous to synchronous reads, you can gain insight into how well the prefetchers are working. This element can be helpful when you are tuning the **num\_ioservers** configuration parameter.

Asynchronous reads are performed by database manager prefetchers.

---

## pool\_async\_xda\_writes - Buffer pool asynchronous XDA data writes monitor element

The number of times a buffer pool data page for an XML storage object (XDA) was physically written to disk by either an asynchronous page cleaner, or a prefetcher. A prefetcher may have written dirty pages to disk to make space for the pages being prefetched.

Table 789. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 790. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 791. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-

**Usage** You can use this element with the **pool\_xda\_writes** monitor element to calculate the number of physical write requests that were performed synchronously on XML storage object data pages (that is, physical data page writes that were performed by database manager agents on XML data). Use the following formula:

$$\text{pool\_xda\_writes} - \text{pool\_async\_xda\_writes}$$

By comparing the ratio of asynchronous to synchronous writes, you can gain insight into how well the buffer pool page cleaners are performing. This ratio can be helpful when you are tuning the **num\_iocleaners** configuration parameter.

---

## pool\_config\_size - Configured Size of Memory Pool

The internally configured size of a memory pool in DB2 database system.

Table 792. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	memory_pool	Basic
Database	memory_pool	Basic
Application	memory_pool	Basic

Table 793. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_dbmemuse	-
Connection	event_connmemuse	-

**Usage** To track system memory usage, use this value in conjunction with *pool\_cur\_size*, *pool\_id*, and *pool\_watermark*.

To see if a memory pool is nearly full, compare *pool\_config\_size* to *pool\_cur\_size*. For example, assume that the utility heap is too small. You can diagnose this specific problem by taking snapshots at regular intervals, and looking in the utility heap section of the snapshot output. If required, the *pool\_cur\_size* might be allowed to exceed the *pool\_config\_size* to prevent an out of memory failure. If this occurs very infrequently, no further action is likely required. However if *pool\_cur\_size* is consistently close to or larger than *pool\_config\_size*, you might consider increasing the size of the utility heap.

---

## pool\_cur\_size - Current Size of Memory Pool

The current size of a memory pool.

Table 794. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	memory_pool	Basic
Database	memory_pool	Basic
Application	memory_pool	Basic

Table 795. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_dbmemuse	-
Connection	event_connmemuse	-

**Usage** To track system memory usage, use this value in conjunction with *pool\_config\_size*, *pool\_id*, and *pool\_watermark*.

To see if a memory pool is nearly full, compare *pool\_config\_size* to *pool\_cur\_size*. For example, assume that the utility heap is too small. You can diagnose this specific problem by taking snapshots at regular intervals, and looking in the utility heap section of the snapshot output. If the value of *pool\_cur\_size* is consistently close to *pool\_config\_size*, you may want to consider increasing the size of the utility heap.

## pool\_data\_l\_reads - Buffer pool data logical reads monitor element

The number of data pages which have been requested from the buffer pool (logical) for regular and large table spaces.

*Table 796. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

*Table 797. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 798. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement

## Usage

This count includes accesses to data that is:

- Already in the buffer pool when the database manager needs to process the page.
- Read into the buffer pool before the database manager can process the page.

Use the **pool\_data\_l\_reads** and **pool\_data\_p\_reads** monitor elements to calculate the overall data page hit ratio for the buffer pool using the following formula:

$$1 - ((\text{pool\_data\_p\_reads} - \text{pool\_async\_data\_reads}) / \text{pool\_data\_l\_reads})$$

Increasing buffer pool size will generally improve the hit ratio, but you will reach a point of diminishing return. Ideally, if you could allocate a buffer pool large enough to store your entire database, then once the system is up and running you would get a hit ratio of 100%. However, this is unrealistic in most cases. The significance of the hit ratio really depends on the size of your data, and the way it is accessed. A very large database where data is accessed evenly would have a poor hit ratio. There is little you can do with very large tables.

To improve hit ratios for smaller, frequently accessed tables and indexes, assign them to individual buffer pools.

---

## pool\_data\_p\_reads - Buffer pool data physical reads monitor element

Indicates the number of data pages read in from the table space containers (physical) for regular and large table spaces.

Table 799. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE

Table 799. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 800. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 801. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE

Table 801. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement

## Usage

Use this element with the **pool\_data\_l\_reads** and **pool\_async\_data\_reads** monitor elements to calculate the number of physical reads that were performed synchronously (that is, physical data page reads that were performed by database manager agents). Use the following formula:

$$1 - ((\text{pool\_data\_p\_reads} + \text{pool\_index\_p\_reads}) - (\text{pool\_async\_data\_reads} + \text{pool\_async\_index\_reads})) / (\text{pool\_data\_l\_reads} + \text{pool\_index\_l\_reads})$$

By comparing the ratio of asynchronous to synchronous reads, you can gain insight into how well the prefetchers are working. This information can be helpful when you are tuning the **num\_ioservers** configuration parameter.

## pool\_data\_writes - Buffer pool data writes monitor element

The number of times a buffer pool data page was physically written to disk.

Table 802. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 802. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 803. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 804. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-

## Usage

If a buffer pool data page is written to disk for a high percentage of the value of the **pool\_data\_p\_reads** monitor element, you may be able to improve performance by increasing the number of buffer pool pages available for the database.

A buffer pool data page is written to disk for the following reasons:

- To free a page in the buffer pool so another page can be read

- To flush the buffer pool

The system does not always write a page to make room for a new one. If the page has not been updated, it can simply be replaced. This replacement is not counted for this element.

The data page can be written by an asynchronous page-cleaner agent before the buffer pool space is required, as reported by the **pool\_async\_data\_writes** monitor element. These asynchronous page writes are included in the value of this element in addition to synchronous page writes.

When calculating this percentage, disregard the number of physical reads required to initially fill the buffer pool. To determine the number of pages written:

1. Run your application (to load the buffer).
2. Note the value of this element.
3. Run your application again.
4. Subtract the value recorded in step 2 from the new value of this element.

In order to prevent the buffer pool from being deallocated between the runnings of your application, you should do one of the following:

- Activate the database with the ACTIVATE DATABASE command.
- Have an idle application connected to the database.

If all applications are updating the database, increasing the size of the buffer pool may not have much impact on performance since most of the buffer pool pages contain updated data, which must be written to disk. However, if the updated pages can be used by other units of work before being written out, the buffer pool can save a write and a read, which will improve your performance.

---

## pool\_drty\_pg\_steal\_clns - Buffer pool victim page cleaners triggered monitor element

The number of times a page cleaner was invoked because a synchronous write was needed during the victim buffer replacement for the database.

*Table 805. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE

*Table 806. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool

For snapshot monitoring, this counter can be reset.

*Table 807. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-



## Usage

Using the following formula, you may calculate what percentage of all cleaner invocations are represented by this element:

$$\frac{\text{pool\_drty\_pg\_steal\_clns}}{(\text{pool\_drty\_pg\_steal\_clns} + \text{pool\_drty\_pg\_thrsh\_clns} + \text{pool\_lsln\_gap\_clns})}$$

If this ratio is low, it may indicate that you have defined too many page cleaners. If your **chnppgs\_thresh** configuration parameter is set too low, you may be writing out pages that you will dirty later. Aggressive cleaning defeats one purpose of the buffer pool, that is to defer writing to the last possible moment.

If this ratio is high, it may indicate that you have not defined enough page cleaners. Not having enough page cleaners increases recovery time after failures.

When the DB2\_USE\_ALTERNATE\_PAGE\_CLEANING registry variable is OFF:

- The **pool\_drty\_pg\_steal\_clns** monitor element is inserted into the monitor stream.
- The **pool\_drty\_pg\_steal\_clns** monitor element counts the number of times a page cleaner was invoked because a synchronous write was needed during the victim buffer replacement for the database.

When the DB2\_USE\_ALTERNATE\_PAGE\_CLEANING registry variable is ON:

- The **pool\_drty\_pg\_steal\_clns** monitor element inserts 0 into the monitor stream.
- There is no explicit triggering of the page cleaners when a synchronous write is needed during victim buffer replacement. To determine whether or not the right number of page cleaners is configured for the database or for specific buffer pools, please refer to the **pool\_no\_victim\_buffer** monitor element.

**Note:** Although dirty pages are written out to disk, the pages are not removed from the buffer pool right away, unless the space is needed to read in new pages.

---

## pool\_drty\_pg\_thrsh\_clns - Buffer pool threshold cleaners triggered monitor element

The number of times a page cleaner was invoked because a buffer pool had reached the dirty page threshold criterion for the database.

*Table 808. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE

*Table 809. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 810. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

**Usage** The threshold is set by the **chnpggs\_thresh** configuration parameter. It is a percentage applied to the buffer pool size. When the number of dirty pages in the pool exceeds this value, the cleaners are triggered.

If the **chnpggs\_thresh** configuration parameter value is set too low, pages might be written out too early, requiring them to be read back in. If it is set too high, then too many pages may accumulate, requiring users to write out pages synchronously.

When the DB2\_USE\_ALTERNATE\_PAGE\_CLEANING registry variable is OFF:

- The **pool\_drty\_pg\_thrsh\_clns** monitor element is inserted into the monitor stream.
- The **pool\_drty\_pg\_thrsh\_clns** monitor element counts the number of times a page cleaner was invoked because a buffer pool had reached the dirty page threshold criterion for the database.

When the DB2\_USE\_ALTERNATE\_PAGE\_CLEANING registry variable is ON:

- The **pool\_drty\_pg\_thrsh\_clns** monitor element inserts 0 into the monitor stream.
- Page cleaners are always active, attempting to ensure there are sufficient free buffers for victims available instead of waiting to be triggered by the criterion value.

## pool\_id - Memory Pool Identifier

The type of memory pool.

Table 811. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	memory_pool	Basic
Database	memory_pool	Basic
Application	memory_pool	Basic

Table 812. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_dbmemuse	-
Connection	event_connmemuse	-

### Usage

To track system memory usage, use this value in conjunction with **pool\_max\_size**, **pool\_cur\_size**, and **pool\_watermark**.

Use **pool\_id** to identify the memory pools discussed in the system monitor output. The various memory pool identifiers can be found in `sqlmon.h`. Under normal operating conditions, one or more of each of the following pools can be expected.

API Constant	Description
SQLM_HEAP_APPLICATION	Application Heap
SQLM_HEAP_DATABASE	Database Heap
SQLM_HEAP_LOCK_MGR	Lock Manager Heap
SQLM_HEAP_UTILITY	Backup/Restore/Utility Heap
SQLM_HEAP_STATISTICS	Statistics Heap
SQLM_HEAP_PACKAGE_CACHE	Package Cache Heap
SQLM_HEAP_CAT_CACHE	Catalog Cache Heap
SQLM_HEAP_MONITOR	Database Monitor Heap
SQLM_HEAP_STATEMENT	Statement Heap
SQLM_HEAP_FCMBP	FCMBP Heap
SQLM_HEAP_IMPORT_POOL	Import Pool
SQLM_HEAP_OTHER	Other Memory
SQLM_HEAP_BP	Buffer Pool Heap
SQLM_HEAP_APPL_SHARED	Applications Shared Heap
SQLM_HEAP_SHARED_SORT	Sort Shared Heap

## pool\_index\_l\_reads - Buffer pool index logical reads monitor element

Indicates the number of index pages which have been requested from the buffer pool (logical) for regular and large table spaces.

*Table 813. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE

Table 813. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 814. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 815. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement

## Usage

This count includes accesses to index pages that are:

- Already in the buffer pool when the database manager needs to process the page
- Read into the buffer pool before the database manager can process the page.

In conjunction with **pool\_index\_p\_reads**, you can calculate the index page hit ratio for the buffer pool using the following formula:

$$1 - ((\text{pool\_index\_p\_reads} - \text{pool\_async\_index\_reads}) / \text{pool\_index\_l\_reads})$$

Use the **pool\_data\_l\_reads** and **pool\_data\_p\_reads** monitor elements to calculate the overall data page hit ratio for the buffer pool using the following formula:

$$1 - ((\text{pool\_data\_p\_reads} - \text{pool\_async\_data\_reads}) / \text{pool\_data\_l\_reads})$$

If the hit ratio is low, increasing the number of buffer pool pages may improve performance.

---

## pool\_index\_p\_reads - Buffer pool index physical reads monitor element

Indicates the number of index pages read in from the table space containers (physical) for regular and large table spaces.

*Table 816. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 817. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 818. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement

## Usage

In conjunction with the **pool\_index\_l\_reads** monitor element, you can calculate the index page hit ratio for the buffer pool using the following formula:

$$1 - ((\text{pool\_index\_p\_reads} - \text{pool\_async\_index\_reads}) / \text{pool\_index\_l\_reads})$$

## pool\_index\_writes - Buffer pool index writes monitor element

Indicates the number of times a buffer pool index page was physically written to disk.

Table 819. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 819. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 820. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 821. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Tablespaces	event_tablespaces	-

Table 821. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

## Usage

Like a data page, a buffer pool index page is written to disk for the following reasons:

- To free a page in the buffer pool so another page can be read
- To flush the buffer pool

The system does not always write a page to make room for a new one. If the page has not been updated, it can simply be replaced. This replacement is not counted for this element.

The index page can be written by an asynchronous page-cleaner agent before the buffer pool space is required. These asynchronous index page writes are included in the value of this element in addition to synchronous index page writes (see the **pool\_async\_index\_writes** monitor element).

If a buffer pool index page is written to disk for a high percentage of the value of the **pool\_index\_p\_reads** monitor element, you may be able to improve performance by increasing the number of buffer pool pages available for the database.

When calculating this percentage, disregard the number of physical reads required to initially fill the buffer pool. To determine the number of pages written:

1. Run your application (to load the buffer).
2. Note the value of this element.
3. Run your application again.
4. Subtract the value recorded in step 2 from the new value of this element.

In order to prevent the buffer pool from being deallocated between the runnings of your application, you should do one of the following:

- Activate the database with the **ACTIVATE DATABASE** command.
- Have an idle application connected to the database.

If all applications are updating the database, increasing the size of the buffer pool may not have much impact on performance, since most of the pages contain updated data which must be written to disk.

---

## pool\_lsn\_gap\_clns - Buffer pool log space cleaners triggered monitor element

The number of times a page cleaner was invoked because the logging space used had reached a predefined criterion for the database.

Table 822. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE



Table 823. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 824. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

## Usage

This element can be used to help evaluate whether you have enough space for logging, and whether you need more log files or larger log files.

The page cleaning criterion is determined by the setting for the **softmax** configuration parameter. Page cleaners are triggered if the oldest page in the buffer pool contains an update described by a log record that is older than the current log position by the criterion value.

When the DB2\_USE\_ALTERNATE\_PAGE\_CLEANING registry variable is OFF:

- The **pool\_lsn\_gap\_clns** monitor element is inserted into the monitor stream.
- Page cleaners are triggered if the oldest page in the buffer pool contains an update described by a log record that is older than the current log position by the criterion value.

When the DB2\_USE\_ALTERNATE\_PAGE\_CLEANING registry variable is ON:

- The **pool\_lsn\_gap\_clns** monitor element inserts 0 into the monitor stream.
- Page cleaners write pages proactively instead of waiting to be triggered by the criterion value.

---

## pool\_no\_victim\_buffer - Buffer pool no victim buffers monitor element

Number of times an agent did not have a preselected victim buffer available.

Table 825. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE

Table 826. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Tablespace	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 827. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespace	event_tablespace	-

**Usage** This element can be used to help evaluate whether you have enough page cleaners for a given buffer pool when using proactive page cleaning.

When the DB2\_USE\_ALTERNATE\_PAGE\_CLEANING registry variable is ON, the pool\_no\_victim\_buffer element counts the number of times that an agent did not find a preselected victim buffer available for immediate use, and was forced to search the buffer pool for a suitable victim buffer.

If the value of pool\_no\_victim\_buffer element is high relative to the number of logical reads in the buffer pool, then the DB2 database system is having difficulty ensuring that sufficient numbers of good victims are available for use. Increasing the number of page cleaners will increase the ability of DB2 to provide preselected victim buffers.

When the DB2\_USE\_ALTERNATE\_PAGE\_CLEANING registry variable is OFF, the pool\_no\_victim\_buffer element has no predictive value, and can be safely ignored. In this configuration, the DB2 database system does not attempt to ensure that agents have preselected victim buffers available to them, so most accesses to the buffer pool will require that the agent search the buffer pool to find a victim buffer.

## pool\_read\_time - Total buffer pool physical read time monitor element

Indicates the total amount of time spent reading in data and index pages from the table space containers (physical) for all types of table spaces. This value is given in milliseconds.

Table 828. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 828. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 829. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 830. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Database	event_db	-
Tablespaces	event_tablespaces	-
Connection	event_conn	-

## Usage

You can use this element with **pool\_data\_p\_reads** and **pool\_index\_p\_reads** monitor elements to calculate the average page-read time. This average is important since it may indicate the presence of an I/O wait, which in turn may indicate that you should be moving data to a different device.

At the database and table space levels, this element includes the value of the **pool\_async\_read\_time** monitor element.

---

## pool\_secondary\_id - Memory Pool Secondary Identifier

An additional identifier to help determine the memory pool for which monitor data is returned.

### Element identifier

pool\_secondary\_id

### Element type

Information

Table 831. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	memory_pool	Basic
Database	memory_pool	Basic
Application	memory_pool	Basic

Table 832. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_dbmemuse	-
Connection	event_connmemuse	-

**Usage** Use together with pool\_id to determine the memory pool for which monitor data is returned. Data for pool\_secondary\_id only appears when necessary. For example, it appears when the pool\_id indicated is Buffer Pool Heap to determine which buffer pool the monitor data relates to.

When a database is created, it has a default buffer pool, called IBMDEFAULTBP, with a size determined by the platform. This buffer pool has a secondary id of "1". In addition to this buffer pool and any buffer pools that you create, a set of system buffer pools are created by default, each corresponding to a different page size. IDs for these buffer pools can appear in snapshots for pool\_secondary\_id:

- System 32k buffer pool
- System 16k buffer pool
- System 8k buffer pool
- System 4k buffer pool

---

## pool\_temp\_data\_l\_reads - Buffer pool temporary data logical reads monitor element

Indicates the number of data pages which have been requested from the buffer pool (logical) for temporary table spaces.

Table 833. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE

Table 833. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 834. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 835. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE

Table 835. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Database	event_db	-
Tablespaces	event_tablespaces	-
Connection	event_conn	-
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement

## Usage

In conjunction with the **pool\_temp\_data\_p\_reads** element, a calculation for the data page hit ratio for buffer pools located in temporary table spaces can be made using the following formula:

$$1 - (\text{pool\_temp\_data\_p\_reads} / \text{pool\_temp\_data\_l\_reads})$$

The overall buffer pool hit ratio can be calculated as follows:

$$1 - ((\text{pool\_data\_p\_reads} + \text{pool\_xda\_p\_reads} + \text{pool\_index\_p\_reads} + \text{pool\_temp\_data\_p\_reads} + \text{pool\_temp\_xda\_p\_reads} + \text{pool\_temp\_index\_p\_reads}) / (\text{pool\_data\_l\_reads} + \text{pool\_xda\_l\_reads} + \text{pool\_index\_l\_reads} + \text{pool\_temp\_data\_l\_reads} + \text{pool\_temp\_xda\_l\_reads} + \text{pool\_temp\_index\_l\_reads})) * 100\%$$

This calculation takes into account all of the pages (index and data) that are cached by the buffer pool.

---

## pool\_temp\_data\_p\_reads - Buffer pool temporary data physical reads monitor element

Indicates the number of data pages read in from the table space containers (physical) for temporary table spaces.

Table 836. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 836. Table Function Monitoring Information (continued)

Table Function	Monitor Element	Collection Level
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE	
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE	
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE	
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE	
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE	
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE	
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE	
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE	

Table 837. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 838. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement

## Usage

The functionality to record buffer pool information at the statement level is supported for API and CLP snapshot requests.

In conjunction with the **pool\_temp\_data\_l\_reads** element, a calculation for the data page hit ratio for buffer pools located in temporary table spaces can be made using the following formula:

$$1 - (\text{pool\_temp\_data\_p\_reads} / \text{pool\_temp\_data\_l\_reads})$$

---

## pool\_temp\_index\_l\_reads - Buffer pool temporary index logical reads monitor element

Indicates the number of index pages which have been requested from the buffer pool (logical) for temporary table spaces.

*Table 839. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE



Table 839. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 840. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 841. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement

## Usage

Use this element, in conjunction with the **pool\_temp\_index\_p\_reads** element, to calculate the index page hit ratio for buffer pools located in temporary table spaces, using the following formula:

$$1 - (\text{pool\_temp\_index\_p\_reads} / \text{pool\_temp\_index\_l\_reads})$$

---

## pool\_temp\_index\_p\_reads - Buffer pool temporary index physical reads monitor element

Indicates the number of index pages read in from the table space containers (physical) for temporary table spaces.

Table 842. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 843. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 844. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement

## Usage

Use this element, in conjunction with the **pool\_temp\_index\_l\_reads** element, to calculate the index page hit ratio for buffer pools located in temporary table spaces, using the following formula:

$$1 - (\text{pool\_temp\_index\_p\_reads} / \text{pool\_temp\_index\_l\_reads})$$

## pool\_temp\_xda\_l\_reads - Buffer pool temporary XDA data logical reads monitor element

Indicates the number of pages for XML storage object (XDA) data which have been requested from the buffer pool (logical) for temporary table spaces.

Table 845. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 845. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 846. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 847. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement

## Usage

You can use the `pool_temp_xda_l_reads` monitor element in conjunction with `pool_temp_xda_p_reads`, `pool_temp_data_l_reads`, and `pool_temp_data_p_reads`

monitor elements to calculate the data page hit ratio for buffer pools located in temporary table spaces by using the following formula:

$$1 - \left( \frac{\text{pool\_temp\_data\_p\_reads} + \text{pool\_temp\_xda\_p\_reads}}{\text{pool\_temp\_data\_l\_reads} + \text{pool\_temp\_xda\_l\_reads}} \right)$$

## pool\_temp\_xda\_p\_reads - Buffer pool temporary XDA data physical reads monitor element

Indicates the number of pages for XML storage object (XDA) data read in from the table space containers (physical) for temporary table spaces.

*Table 848. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

*Table 849. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool

Table 849. Snapshot Monitoring Information (continued)

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 850. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement

## Usage

You can use the **pool\_temp\_xda\_p\_reads** monitor element in conjunction with **pool\_temp\_xda\_l\_reads**, **pool\_temp\_data\_l\_reads**, and **pool\_temp\_data\_p\_reads** monitor elements to calculate the data page hit ratio for buffer pools located in temporary table spaces by using the following formula:

$$1 - ((\text{pool\_temp\_data\_p\_reads} + \text{pool\_temp\_xda\_p\_reads}) / (\text{pool\_temp\_data\_l\_reads} + \text{pool\_temp\_xda\_l\_reads}))$$

## pool\_watermark - Memory Pool Watermark

The largest size of a memory pool since its creation.

### Element identifier

pool\_watermark

### Element type

Information

Table 851. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	memory_pool	Basic
Database	memory_pool	Basic
Application	memory_pool	Basic

Table 852. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_dbmemuse	-
Connection	event_connmemuse	-

**Usage** On continuously running systems, you can use the *pool\_watermark* and *pool\_config\_size* elements together to predict potential memory problems.

For example, take a snapshot at regular intervals (for instance, daily), and examine the *pool\_watermark* and *pool\_config\_size* values. If you observe that the value of *pool\_watermark* is becoming increasingly close to *pool\_config\_size* (a premature indication of potential future memory-related problems), this may indicate that you should increase the size of the memory pool.

## pool\_write\_time - Total buffer pool physical write time monitor element

Provides the total amount of time spent physically writing data or index pages from the buffer pool to disk. Elapsed time is given in milliseconds.

Table 853. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

Table 853. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 854. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 855. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-

## Usage

Use this element with **buffer\_pool\_data\_writes** and **pool\_index\_writes** monitor elements to calculate the average page-write time. This average is important since it may indicate the presence of an I/O wait, which in turn may indicate that you should be moving data to a different device.

At the database and table space levels, this element includes the value of the **pool\_async\_write\_time** monitor element.

---

## pool\_xda\_l\_reads - Buffer pool XDA data logical reads monitor element

Indicates the number of data pages for XML storage objects (XDAs) which have been requested from the buffer pool (logical) for regular and large table spaces.

Table 856. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE



Table 856. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 857. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 858. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement

## Usage

This count includes accesses to data that is:

- Already in the buffer pool when the database manager needs to process the page
- Read into the buffer pool before the database manager can process the page.

Use the **pool\_xda\_l\_reads**, **pool\_xda\_p\_reads**, **pool\_data\_l\_reads**, and **pool\_data\_p\_reads** monitor elements to calculate the data page hit ratio for the buffer pool by using the following formula:

$$1 - ((\text{pool\_data\_p\_reads} + \text{pool\_xda\_p\_reads}) / (\text{pool\_data\_l\_reads} + \text{pool\_xda\_l\_reads}))$$

The overall buffer pool hit ratio can be calculated as follows:

$$1 - ((\text{pool\_data\_p\_reads} + \text{pool\_xda\_p\_reads} + \text{pool\_index\_p\_reads} + \text{pool\_temp\_data\_p\_reads} + \text{pool\_temp\_xda\_p\_reads} + \text{pool\_temp\_index\_p\_reads}) / (\text{pool\_data\_l\_reads} + \text{pool\_xda\_l\_reads} + \text{pool\_index\_l\_reads} + \text{pool\_temp\_data\_l\_reads} + \text{pool\_temp\_xda\_l\_reads} + \text{pool\_temp\_index\_l\_reads})) * 100\%$$

This calculation takes into account all of the pages (index and data) that are cached by the buffer pool.

Increasing buffer pool size will generally improve the hit ratio, but you will reach a point of diminishing return. Ideally, if you could allocate a buffer pool large enough to store your entire database, then once the system is up and running you would get a hit ratio of 100%. However, this is unrealistic in most cases. The significance of the hit ratio depends on the size of your data, and the way it is accessed. A very large database where data is accessed evenly would have a poor hit ratio. There is little you can do with very large tables. In such case, you would focus your attention on smaller, frequently accessed tables, and on the indexes.

## pool\_xda\_p\_reads - Buffer pool XDA data physical reads monitor element

Indicates the number of data pages for XML storage objects (XDAs) read in from the table space containers (physical) for regular and large table spaces.

*Table 859. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

*Table 860. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool
Application	stmt	Buffer Pool
Dynamic SQL	dynsql	Buffer Pool, Statement

For snapshot monitoring, this counter can be reset.

Table 861. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Tablespaces	event_tablespaces	-
Connection	event_conn	-
Statement	event_stmt	-
Activities	event_activity	Buffer Pool, Statement

## Usage

Use the **pool\_async\_xda\_reads** and **pool\_xda\_p\_reads** monitor elements to calculate the number of physical reads that were performed synchronously on XML storage object data pages (that is, physical data page reads that were performed by database manager agents on XML data). Use the following formula:

$$\text{pool\_xda\_p\_reads} - \text{pool\_async\_xda\_reads}$$

By comparing the ratio of asynchronous to synchronous reads, you can gain insight into how well the prefetchers are working. This element can be helpful when you are tuning the **num\_ioservers** configuration parameter.

Use the **pool\_xda\_l\_reads**, **pool\_xda\_p\_reads**, **pool\_data\_l\_reads**, and **pool\_data\_p\_reads** monitor elements to calculate the data page hit ratio for the buffer pool by using the following formula:

$$1 - \left( \frac{\text{pool\_data\_p\_reads} + \text{pool\_xda\_p\_reads}}{\text{pool\_data\_l\_reads} + \text{pool\_xda\_l\_reads}} \right)$$

---

## pool\_xda\_writes - Buffer pool XDA data writes monitor element

Indicates the number of times a buffer pool data page for an XML storage object (XDA) was physically written to disk.

Table 862. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 862. Table Function Monitoring Information (continued)

Table Function	Monitor Element	Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS	BASE
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS	BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS	BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS	BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS	BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS	BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS	BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS	BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS	BASE

Table 863. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

Table 864. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Tablespaces	event_tablespace	-

Table 864. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

## Usage

This monitor element helps you to assess whether performance may be improved by increasing the number of buffer pool pages available for the database. For databases containing XML data, you should consider the ratio of buffer pool page writes to buffer pool page reads both for XML data (using the **pool\_xda\_writes** and the **pool\_xda\_p\_reads** monitor elements) and for relational data types (using the **pool\_data\_writes** and the **pool\_data\_p\_reads** monitor elements).

Use the **pool\_xda\_l\_reads**, **pool\_xda\_p\_reads**, **pool\_data\_l\_reads**, and **pool\_data\_p\_reads** monitor elements to calculate the data page hit ratio for the buffer pool by using the following formula:

$$1 - \left( \frac{\text{pool\_data\_p\_reads} + \text{pool\_xda\_p\_reads}}{\text{pool\_data\_l\_reads} + \text{pool\_xda\_l\_reads}} \right)$$

---

## post\_shrthreshold\_hash\_joins - Post threshold hash joins

The total number of hash joins that were throttled back by the sort memory throttling algorithm. A throttled hash join is a hash join that was granted less memory than requested by the sort memory manager.

Table 865. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	-

For snapshot monitoring, this counter can be reset.

Table 866. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

A hash join is throttled back when the memory allocation from the shared sort heap is close to the limit set by database configuration parameter *sheapthres\_shr*. This throttling will significantly reduce the number of overflows over *sheapthres\_shr* limit in a system that is not properly configured. The data reported in this element only reflects hash joins using memory allocated from the shared sort heap.

---

## post\_shrthreshold\_sorts - Post shared threshold sorts monitor element

The total number of sorts that were throttled back by the sort memory throttling algorithm. A throttled sort is a sort that was granted less memory than requested by the sort memory manager. A sort is throttled back when the memory allocation for sorts is close to the limit set by database configuration parameter *sheapthres\_shr*. This throttling will significantly reduce the number of overflows over *sheapthres\_shr* limit in a system that is not properly configured. The data reported in this element only reflects sorts using memory allocated from the shared sort heap.

Table 867. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 868. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Sort

For snapshot monitoring, this counter can be reset.

Table 869. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-

---

## post\_threshold\_hash\_joins - Hash Join Threshold

The total number of times that a hash join heap request was limited due to concurrent use of shared or private sort heap space.

### Element identifier

post\_threshold\_hash\_joins

### Element type

counter

Table 870. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

For snapshot monitoring, this counter can be reset.

**Usage** If this value is large (greater than 5% of hash\_join\_overflows), the sort heap threshold should be increased.

---

## post\_threshold\_olap\_funcs - OLAP Function Threshold monitor element

The number of OLAP functions that have requested a sort heap after the sort heap threshold has been exceeded.

Table 871. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

For snapshot monitoring, this counter can be reset.

### Usage

Sorts, hash joins, and OLAP functions are examples of operations which utilize a sort heap. Under normal conditions, the database manager will allocate sort heap using the value specified by the sortheap configuration parameter. If the amount of memory allocated to sort heaps exceeds the sort heap threshold (sheapthres configuration parameter), the database manager will allocate subsequent sort heaps using a value less than that specified by the sortheap configuration parameter.

OLAP functions which start after the sort heap threshold has been reached may not receive an optimum amount of memory to execute.

To improve sort, hash join, OLAP function performance, and overall system performance, modify the sort heap threshold and sort heap size configuration parameters.

If this element's value is high, increase the sort heap threshold (sheapthres).

---

## post\_threshold\_sorts - Post threshold sorts monitor element

The number of sorts that have requested heaps after the sort heap threshold has been exceeded.



Table 872. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 873. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Sort

For snapshot monitoring, this counter can be reset.

Table 874. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## Usage

Under normal conditions, the database manager will allocate sort heap using the value specified by the **sortheap** configuration parameter. If the amount of memory allocated to sort heaps exceeds the sort heap threshold (**sheapthres** configuration parameter), the database manager will allocate sort heap using a value less than that specified by the **sortheap** configuration parameter.

Each active sort on the system allocates memory, which may result in sorting taking up too much of the system memory available. Sorts that start after the sort heap threshold has been reached may not receive an optimum amount of memory to execute, but, as a result, the entire system may benefit. By modifying the sort heap threshold and sort heap size configuration parameters, sort operation performance and overall system performance can be improved. If this element's value is high, you can:

- Increase the sort heap threshold (**sheapthres**) or,
- Adjust applications to use fewer or smaller sorts via SQL query changes.

---

## prefetch\_wait\_time - Time waited for prefetch monitor element

The time an application spent waiting for an I/O server (prefetcher) to finish loading pages into the buffer pool. The value is given in milliseconds.

*Table 875. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Application	appl	Buffer Pool

*Table 876. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** This element can be used to experiment with changing the number of I/O servers, and I/O server sizes.

---

## prep\_time - Preparation time monitor element

Time in milliseconds required to prepare an SQL statement if the activity is an SQL statement.

*Table 877. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

*Table 878. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

## Usage

This element can be used to identify how much of the activity's total lifetime was spent preparing the SQL statement, if this was an SQL activity.

---

### prep\_time\_best - Statement best preparation time monitor element

The shortest amount of time in milliseconds that was required to prepare a specific SQL statement.

Table 879. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Dynamic SQL	dynsql	Basic

## Usage

Use this value in conjunction with **prep\_time\_worst** to identify SQL statements that are expensive to compile.

---

### prep\_time\_worst - Statement worst preparation time monitor element

The longest amount of time in milliseconds that was required to prepare a specific SQL statement.

Table 880. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Dynamic SQL	dynsql	Basic

## Usage

Use this value in conjunction with **prep\_time\_best** to identify SQL statements that are expensive to compile.

---

### prev\_uow\_stop\_time - Previous Unit of Work Completion Timestamp

This is the time the unit of work completed.

#### Element identifier

prev\_uow\_stop\_time

#### Element type

timestamp

Table 881. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Unit of Work, Timestamp
DCS Application	dcs_appl	Unit of Work, Timestamp

**Usage** You may use this element with *uow\_stop\_time* to calculate the total elapsed time between COMMIT/ROLLBACK points, and with *uow\_start\_time* to calculate the time spent in the application between units of work. The time of one of the following:

- For applications currently within a unit of work, this is the time that the latest unit of work completed.
- For applications not currently within a unit of work (the application has completed a unit of work, but not yet started a new one), this is the stop time of the last unit of work that completed prior to the one that just completed. The stop time of the one just completed is indicated `uow_stop_time`.
- For applications within their first unit of work, this is the database connection request completion time.

---

## **priv\_workspace\_num\_overflows - Private Workspace Overflows**

The number of times that the private workspaces overflowed the bounds of its allocated memory.

**Note:** This monitor element has been deprecated. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

*Table 882. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

*Table 883. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** Use this element with `priv_workspace_size_top` to determine whether the size of the private workspace needs to be increased to avoid overflowing. Overflows of the private workspace may cause performance degradation as well as out of memory errors from the other heaps allocated out of agent private memory.

At the database level, the element reported will be from the same private workspace as that which was reported as having the same Maximum Private Workspace size. At the application level, it is the number of overflows for the workspace of every agent that have serviced the current application.

---

## **priv\_workspace\_section\_inserts - Private Workspace Section Inserts**

Inserts of SQL sections by an application into the private workspace.

**Note:** This monitor element has been deprecated. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

Table 884. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 885. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** The working copy of executable sections are stored in the private workspace.

This counter indicates when a copy was not available and had to be inserted. At the database level, it is the cumulative total of all inserts for every application across all private workspaces in the database. At the application level, it is the cumulative total of all inserts for all sections in the private workspace for this application.

In a concentrator environment where agents are being associated with different applications, additional private workspace inserts may be required as a result of a new agent not having the required section available in its private workspace.

---

## priv\_workspace\_section\_lookups - Private Workspace Section Lookups

Lookups of SQL sections by an application in its agents' private workspace.

**Note:** This monitor element has been deprecated. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

Table 886. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 887. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** Each application has access to the private workspace of the agent working for it.

This counter indicates how many times the private workspace was accessed in order to locate a specific section for an application. At the

database level, it is the cumulative total of all lookups for every application across all private workspaces in the database. At the application level, it is the cumulative total of all lookups for all sections in the private workspace for this application.

You can use this element in conjunction with Private Workspace Section Inserts to tune the size of the private workspace. The size of the private workspace is controlled by the `applheapsz` configuration parameter.

---

## priv\_workspace\_size\_top - Maximum Private Workspace Size

The largest size reached by the Private Workspace.

**Note:** This monitor element has been deprecated. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

*Table 888. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

*Table 889. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** Each agent has a private workspace that the application it is servicing has access to. This element indicates the maximum number of bytes required from a private workspace by any agent servicing it. At the database level, it is the maximum number of bytes required of all the private workspaces for all agents attached to the current database. At the application level, it is the maximum size from among all of the agents' private workspaces that have serviced the current application.

When the private workspace overflows, memory is temporarily borrowed from other entities in agent private memory. This can result in memory shortage errors from these entities or possibly performance degradation. You can reduce the chance of overflow by increasing `APPLHEAPSZ`.

---

## product\_name - Product Name

Details of the version of the DB2 instance that is running.

*Table 890. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

---

## progress\_completed\_units - Completed Progress Work Units

The number of work units for the current phase which have been completed.

Table 891. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	progress	Basic

The value of this element will typically increase as the utility operates. This element will always be less than or equal to *progress\_total\_units* (if both elements are defined).

**Note:**

1. This element might not be included for all utilities.
2. This element is expressed in units displayed by the *progress\_work\_metric* monitor element.

**Usage** Use this element to determine the amount of completed work within a phase. By itself, this element can be used to monitor the activity of a running utility. This element should constantly increase as the utility executes. If the *progress\_completed\_units* fails to increase over a long period of time then the utility might be stalled.

If *progress\_total\_units* is defined, then this element can be used to calculate the percentage of completed work:

$$\text{percentage complete} = \text{progress\_completed\_units} / \text{progress\_total\_units} * 100$$

## progress\_description - Progress Description

Describes the phase of work.

Table 892. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	progress	Basic

Example values for the load utility include:

- DELETE
- LOAD
- REDO

**Usage** Use this element to obtain a general description of a phase.

## progress\_list\_attr - Current Progress List Attributes

This element describes how to interpret a list of progress elements.

Table 893. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	progress list	Basic

**Usage**

The value for this element is one of the following constants:

- *SQLM\_ELM\_PROGRESS\_LIST\_ATTR\_SERIAL* - The elements in the list are to be interpreted as a set of serial phases meaning that completed work must equal the total work for element *n* before the completed work of element *n+1* is first

updated. This attribute is used to describe progress of a task which consists of a set of serial phases where a phase must fully complete before the next phase begins.

- `SQLM_ELM_PROGRESS_LIST_ATTR_CONCURRENT` - Any element in the progress list can be updated at any time.

Use this element to determine how the elements of a progress list will be updated.

---

## progress\_list\_cur\_seq\_num - Current Progress List Sequence Number

If the utility contains multiple sequential phases, then this element displays the number of the current phase.

Table 894. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	progress_list	Basic

**Usage** Use this element to determine the current phase of a multiphase utility. See “progress\_seq\_num - Progress Sequence Number.”

---

## progress\_seq\_num - Progress Sequence Number

Phase number.

**Note:** The phase number displays only for utilities that consist of multiple phases of execution.

Table 895. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	progress	Basic

**Usage** Use this element to determine the order of phases within a multiphase utility. The utility will execute phases serially in order of increasing progress sequence numbers. The current phase of a multiphase utility can be found by matching the *progress\_seq\_num* with the value of *progress\_list\_current\_seq\_num*.

---

## progress\_start\_time - Progress Start Time

A timestamp representing the start of the phase.

Table 896. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	progress	Basic

**Usage** Use this element to determine when a phase started. This element is omitted if the phase has not yet begun.

---

## progress\_total\_units - Total Progress Work Units

Total amount of work to perform in order for the phase to be complete.



Table 897. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	progress	Basic

Some utilities might not be able to quantify the total work so they will continuously update this element. Other utilities might not be able to provide an estimate for the total work so this element might be omitted entirely.

This element is expressed in units displayed by the *progress\_work\_metric* monitor element.

**Usage** Use this element to determine the total amount of work in the phase. Use this element with *progress\_completed\_units* to calculate the percentage of work completed within a phase:

$$\text{percentage complete} = \text{progress\_completed\_units} / \text{progress\_total\_units} * 100$$

---

## progress\_work\_metric - Progress Work Metric

The metric for interpreting the *progress\_total\_units* and *progress\_completed\_units* elements.

Table 898. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	progress	Basic

Example values include:

- SQLM\_WORK\_METRIC\_BYTES
- SQLM\_WORK\_METRIC\_EXTENTS

**Note:**

1. This element might not be included for all utilities.
2. Values for this element can be found in *sqlmon.h*

**Usage** Use this element to determine what *progress\_total\_units* and *progress\_completed\_units* use as their reporting metric.

---

## pseudo\_deletes - Pseudo deletes monitor element

All keys on pseudo empty pages have been pseudo deleted. This monitor element reports the number of pseudo empty pages that have been deleted.

Table 899. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

---

## pseudo\_empty\_pages - Pseudo empty pages monitor element

All keys on pseudo empty pages have been pseudo deleted. This monitor element reports the number of pages that have been identified as pseudo empty.

Table 900. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

## Usage

**Note:** This monitor element does not report the current number of pseudo empty pages.

## qp\_query\_id - Query patroller query ID monitor element

The query ID assigned to this activity by Query Patroller if the activity is a query. A query ID of 0 indicates that Query Patroller did not assign a query ID to this activity.

**Important:** The qp\_query\_id monitor element is deprecated because it is associated with Query Patroller functionality. With the new workload management features introduced in DB2 Version 9.5, Query Patroller and its related components have been deprecated in Version 9.7 and might be removed in a future release.

Table 901. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

## query\_card\_estimate - Query Number of Rows Estimate

An estimate of the number of rows that will be returned by a query.

Table 902. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement
DCS Statement	dcs_stmt	Statement
Activities	event_activity	-

**Usage** This estimate by the SQL compiler can be compared with the run time actuals.

This element also returns information for the following SQL statements when you are monitoring DB2 Connect.

- INSERT, UPDATE, and DELETE  
Indicates the number of rows affected.
- PREPARE  
Estimate of the number of rows that will be returned. Only collected if the DRDA server is DB2 Database for Linux, UNIX, and Windows, DB2 for VM and VSE, or DB2 for OS/400®.
- FETCH

Set to the number of rows fetched. Only collected if the DRDA server is DB2 for OS/400.

If information is not collected for a DRDA server, then the element is set to zero.

---

## query\_cost\_estimate - Query cost estimate monitor element

Estimated cost for a query, as determined by the SQL compiler. This value is reported in timerons.

*Table 903. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

*Table 904. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement
DCS Statement	dcs_stmt	Statement
Activities	event_activity	-

**Usage** This allows correlation of actual run-time with the compile-time estimates.

This element also returns information for the following SQL statements when you are monitoring DB2 Connect.

- PREPARE  
Represents the relative cost of the prepared SQL statement.
- FETCH  
Contains the length of the row retrieved. Only collected if the DRDA server is DB2 for OS/400.

If information is not collected for a DRDA server, then the element is set to zero.

**Note:** If the DRDA server is DB2 for OS/390<sup>®</sup> and z/OS, this estimate could be higher than  $2^{32} - 1$  (the maximum integer number that can be expressed through an unsigned long variable). In that case, the value returned by the monitor for this element will be  $2^{32} - 1$ .

---

## queue\_assignments\_total - Queue assignments total monitor element

The number of times any connection or activity was assigned to this threshold queue since the last reset.

*Table 905. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_qstats	-

## Usage

This element can be used to determine the number of times any connection or activity was queued in this particular queue in a given period of time determined by the statistics collection interval. This can help to determine the effectiveness of queuing thresholds.

---

### queue\_size\_top - Queue size top monitor element

Highest queue size that has been reached since the last reset.

*Table 906. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_qstats	-

## Usage

Use this element to gauge the effectiveness of queuing thresholds and to detect when queuing is excessive.

---

### queue\_time\_total - Queue time total monitor element

Sum of the times spent in the queue for all connections or activities placed in this queue since the last reset. Units are milliseconds.

*Table 907. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_qstats	-

## Usage

Use this element to gauge the effectiveness of queuing thresholds and to detect when queuing is excessive.

---

### quiescer\_agent\_id - Quiescer Agent Identification

Agent ID of the agent holding a quiesce state.

**Element identifier**

quiescer\_agent\_id

**Element type**

information

*Table 908. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_quiescer	Basic

**Usage** Use this element in conjunction with quiescer\_auth\_id to determine who is responsible for quiescing a table space.

---

### quiescer\_auth\_id - Quiescer User Authorization Identification

Authorization ID of the user holding a quiesce state.

**Element identifier**  
quiescer\_auth\_id

**Element type**  
information

*Table 909. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_quiescer	Basic

**Usage** Use this element to determine who is responsible for quiescing a table space.

---

## quiescer\_obj\_id - Quiescer Object Identification

The object ID of the object that causes a table space to be quiesced.

**Element identifier**  
quiescer\_obj\_id

**Element type**  
information

*Table 910. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_quiescer	Basic

**Usage** Use this element in conjunction with quiescer\_ts\_id and quiescer\_auth\_id to determine who is responsible for quiescing a table space. The value of this element matches a value from column TABLEID of view SYSCAT.TABLES.

---

## quiescer\_state - Quiescer State

The type of quiesce being done (for example, "SHARE", "INTENT TO UPDATE", or "EXCLUSIVE").

**Element identifier**  
quiescer\_state

**Element type**  
information

*Table 911. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_quiescer	Basic

**Usage** The value of this element matches the value of constants SQLB\_QUIESCED\_SHARE, SQLB\_QUIESCED\_UPDATE, or SQLB\_QUIESCED\_EXCLUSIVE from sqlutil.h.

---

## quiescer\_ts\_id - Quiescer Table Space Identification

The table space ID of the object that causes a table space to be quiesced.

**Element identifier**  
quiescer\_ts\_id

**Element type**  
information

*Table 912. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_quiescer	Basic

**Usage** Use this element in conjunction with `quiescer_obj_id` and `quiescer_auth_id` to determine who is responsible for quiescing a table space. The value of this element matches a value from column `TBSPACEID` of view `SYSCAT.TABLES`.

---

## range\_adjustment - Range Adjustment

This value represents the offset into the container array in which a range actually starts.

*Table 913. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_range	Basic

**Usage** This element is applicable only to a DMS table space.

---

## range\_container\_id - Range Container

An integer that uniquely defines a container within a range.

*Table 914. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_range	Basic

**Usage** This element is applicable only to a DMS table space.

---

## range\_end\_stripe - End Stripe

This value represents the number of the last stripe in a range.

*Table 915. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_range	Basic

**Usage** This element is applicable only to a DMS table space.

---

## range\_max\_extent - Maximum Extent in Range

This value represents the maximum extent number that is mapped by a range.

*Table 916. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_range	Basic

**Usage** This element is applicable only to a DMS table space.

---

## range\_max\_page\_number - Maximum Page in Range

This value represents the maximum page number that is mapped by a range.

*Table 917. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_range	Basic

**Usage** This element is applicable only to a DMS table space.

---

## range\_num\_containers - Number of Containers in Range

This value represents the number of containers in the current range.

*Table 918. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_range	Basic

**Usage** This element is applicable only to a DMS table space.

---

## range\_number - Range Number

This value represents the number of a range within the table space map.

*Table 919. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_range	Basic

**Usage** This element is applicable only to a DMS table space.

---

## range\_offset - Range Offset

The offset from stripe 0 of the beginning of the stripe set to which a range belongs.

*Table 920. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_range	Basic

**Usage** This element is applicable only to a DMS table space.

---

## range\_start\_stripe - Start Stripe

This value represents the number of the first stripe in a range.

*Table 921. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_range	Basic

**Usage** This element is applicable only to a DMS table space.

---

## range\_stripe\_set\_number - Stripe Set Number

This value represents the stripe set in which a range resides.

Table 922. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_range	Basic

**Usage** This element is applicable only to a DMS table space.

---

## reclaimable\_space\_enabled - Reclaimable space enabled indicator monitor element

If the table space is enabled for reclaimable storage, then this monitor element returns a value of 1. Otherwise, it returns a value of 0.

Table 923. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

---

## rej\_curs\_blk - Rejected Block Cursor Requests

The number of times that a request for an I/O block at server was rejected and the request was converted to non-blocked I/O.

**Element identifier**

rej\_curs\_blk

**Element type**

counter

Table 924. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

Table 925. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-

**Usage** If there are many cursors blocking data, the communication heap may become full. When this heap is full, an error is not returned. Instead, no more I/O blocks are allocated for blocking cursors. If cursors are unable to block data, performance can be affected.

If a large number of cursors were unable to perform data blocking, you may be able to improve performance by:

- Increasing the size of the *query\_heap* database manager configuration parameter.



---

## rem\_cons\_in - Remote Connections To Database Manager

The current number of connections initiated from remote clients to the instance of the database manager that is being monitored.

Table 926. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

### Usage

Shows the number of connections from remote clients to databases in this instance. This value will change frequently, so you may need to sample it at specific intervals over an extended period of time to get a realistic view of system usage. This number does not include applications that were initiated from the same instance as the database manager.

When used in conjunction with the local\_cons monitor element, these elements can help you adjust the setting of the **max\_coordagents** and **max\_connections** configuration parameters.

---

## rem\_cons\_in\_exec - Remote Connections Executing in the Database Manager

The number of remote applications that are currently connected to a database and are currently processing a unit of work within the database manager instance being monitored.

Table 927. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

### Usage

This number can help you determine the level of concurrent processing occurring on the database manager. This value will change frequently, so you may need to sample it at specific intervals over an extended period of time to get a realistic view of system usage. This number does not include applications that were initiated from the same instance as the database manager.

When used in conjunction with the local\_cons\_in\_exec monitor element, this element can help you adjust the setting of the **max\_coordagents** configuration parameter.

If **max\_coordagents** is set to AUTOMATIC, then you do not need to make any adjustments. If it is not set to AUTOMATIC and if the sum of rem\_cons\_in\_exec and local\_cons\_in\_exec is close to **max\_coordagents**, you should increase the value of **max\_coordagents**.

---

## remote\_lock\_time - Remote Lock Time

This element contains the aggregate amount of time, in milliseconds, that this data source spends in a remote lock from all applications or a single application running on this federated server instance since the start of the federated server instance or the last reset of the database monitor counters, whichever is latest. The response time is measured as the difference between the time the federated server submits a remote lock to the data source, and the time the federated server releases a remote lock at the data source.

Table 928. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Timestamp
Application	appl_remote	Timestamp

For snapshot monitoring, this counter can be reset.

### Usage

Use this element to determine how much actual time is spent at this data source in a remote lock.

---

## remote\_locks - Remote Locks

This element contains a count of the total number of remote locks that the federated server has called at this data source on behalf of any application since the start of the federated server instance or the last reset of the database monitor counters, whichever is latest.

Table 929. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to determine how many remote locks were made remotely at the data source.

---

## reorg\_completion - Reorganization Completion Flag

Table reorganization success indicator, which includes the reclamation of extents from a multidimensional clustering (MDC) table. For partitioned tables, this also indicates the completion status for the data partition.

Table 930. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

**Usage** This element will have a value of 0 if a table or data partition reorganize operation is successful. If a table or data partition reorganize operation is unsuccessful, this element will have a value of -1. Success and failure values are defined in sqlmon.h as follows:

- Success: SQLM\_REORG\_SUCCESS
- Failure: SQLM\_REORG\_FAIL

In the case of an unsuccessful table reorganization, see the history file for any diagnostic information, including warnings and errors. This data can be accessed by using the LIST HISTORY command. For partitioned tables, the completion status is indicated per data partition. If index recreate fails on a partitioned table, the failed status is updated on all data partitions. See the administration notification log for further diagnostic information.

---

## reorg\_current\_counter - Reorganize Progress

A unit of progress that indicates the amount of reorganization that has been completed. The amount of progress this value represents is relative to the value of reorg\_max\_counter, which represents the total amount of table reorganization that is to be done.

*Table 931. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

### Usage

You can determine the percentage of table reorganization that has been completed using the following formula:

$\text{table reorg progress} = \text{reorg\_current\_counter} / \text{reorg\_max\_counter} * 100$

---

## reorg\_end - Table Reorganize End Time

The end time of a table reorganization including a reorganization to reclaim extents from a multidimensional clustering (MDC) table. For partitioned tables, this will also indicate the end time for each data partition reorganization.

*Table 932. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

---

## reorg\_index\_id - Index Used to Reorganize the Table

The index being used to reorganize the table.

*Table 933. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

---

## reorg\_long\_tbspc\_id - Table Space Where Long Objects are Reorganized monitor element

The table space in which any long objects (LONG VARCHAR or LOB data) will be reorganized. For partitioned tables, this is the table space in which each partition's LONG VARCHAR and LOB will be reorganized.

Table 934. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

---

## reorg\_max\_counter - Total Amount of Reorganization

A value that indicates the total amount of work to be done in a reorganization including a reorganization to reclaim extents from multidimensional clustering (MDC) tables. This value can be used with `reorg_current_counter`, which represents the amount of work completed, to determine the progress of a reorganization.

Table 935. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

---

## reorg\_max\_phase - Maximum Reorganize Phase

The maximum number of reorganization phases that will occur during reorganization processing. This applies to classic (offline) reorganizations only. The range of values is 2 to 4 ([SORT], BUILD, REPLACE,[INDEX\_RECREATE]). The value could also indicate the total amount of work to be done in a reorganization to reclaim extents from a multidimensional clustering (MDC) table. When such a reorganization occurs, this value is 3 (SCAN, DRAIN, RELEASE).

Table 936. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

---

## reorg\_phase - Table reorganization phase monitor element

Indicates the reorganization phase of the table. For partitioned tables, this will also indicate the reorganization phase for each data partition. This applies to offline table reorganizations only.

Table 937. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

### Usage

For partitioned tables, the reorganization occurs on a data partition by data partition basis. For classic table reorganization, the following phases are possible (phases are listed with their corresponding defines from `thesqlmon.h` file):

- Sort: `SQLM_REORG_SORT`
- Build: `SQLM_REORG_BUILD`
- Replace: `SQLM_REORG_REPLACE`
- Index Recreate: `SQLM_REORG_INDEX_RECREATE`
- Dictionary Build: `SQLM_REORG_DICT_SAMPLE`

For partitioned tables, the Index Recreate phase for partitioned indexes (if any) might be directly entered after the replace phase for that data partition. The **reorg\_phase** element will indicate the Index Recreate phase only after the successful completion of all prior phases on every data partition.

During XDA object compression, the XML data reorganization phase involves reorganizing the XML storage object of the table. The XML dictionary build phase involves attempting to create a compression dictionary for the XML storage object. For XDA object compression, the following two phases are possible:

- XML Reorg: SQLM\_REORG\_XML\_DATA
- XML Dictionary Build: SQLM\_REORG\_XML\_DICT\_SAMPLE

For partitioned tables, where reclamation of extents is being performed, the following phases are possible:

- Scan: SQLM\_REORG\_SCAN
- Drain: SQLM\_REORG\_DRAIN
- Release: SQLM\_REORG\_RELEASE

---

## reorg\_phase\_start - Reorganize Phase Start Time

The start time of a phase of table reorganization or reclaim reorganization. For partitioned tables, this will also indicate the start time of a reorganization phase for each data partition. During the index recreate phase, data groups for all data partitions are updated at the same time for nonpartitioned indexes.

*Table 938. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

---

## reorg\_rows\_compressed - Rows Compressed

Number of rows compressed in the table during reorganization.

*Table 939. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

**Usage** A running count of the number of rows compressed in the table during reorganization. Some records may never be compressed (if the record size is less than the minimum record length).

It is important to note that this row count does not measure the effectiveness of data compression. It only displays the number of records meeting compression criteria.

---

## reorg\_rows\_rejected\_for\_compression - Rows Rejected for Compression

Number of rows that were not compressed during reorganization due to the record length being less than or equal to the minimum record length.

Table 940. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

**Usage** A record will not be compressed if it is less than or equal to the minimum record length. The number of rows rejected reflects a running count for these records that fail to meet this compression requirement.

---

## reorg\_start - Table Reorganize Start Time

The start time of a table reorganization including a reorganization to reclaim extents from a multidimensional clustering (MDC) table. For partitioned tables, this will also indicate the start time for each data partition reorganization.

Table 941. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

---

## reorg\_status - Table Reorganize Status

The status of an in-place (online) table or a data partition level reorganization. This is not applicable to classic (offline) table reorganizations.

Table 942. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

**Usage** An in-place table or data partition reorganization can be in one of the following states (states are listed with their corresponding defines from `sqlmon.h`):

- Started/Resumed: `SQLM_REORG_STARTED`
- Paused: `SQLM_REORG_PAUSED`
- Stopped: `SQLM_REORG_STOPPED`
- Completed: `SQLM_REORG_COMPLETED`
- Truncate: `SQLM_REORG_TRUNCATE`

An in-place table or data partition reorganization to reclaim extents can be in one of the following states:

- Started: `SQLM_REORG_STARTED`
- Stopped: `SQLM_REORG_STOPPED`
- Completed: `SQLM_REORG_COMPLETED`

---

## reorg\_tbspc\_id - Table Space Where Table or Data partition is Reorganized

The table space in which the table will be reorganized. For partitioned tables, this indicates the table space where each data partition is reorganized.

Table 943. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

## reorg\_type - Table Reorganize Attributes

Table reorganize attribute settings.

Table 944. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

**Usage** The following are possible attribute settings. Each attribute setting is based upon a bit flag value defined in db2ApiDf.h.

- Allow Write Access: DB2REORG\_ALLOW\_WRITE
- Allow Read Access: DB2REORG\_ALLOW\_READ
- Allow No Access: DB2REORG\_ALLOW\_NONE
- Recluster Via Index Scan: DB2REORG\_INDEXSCAN
- Reorg Long Field LOB Data: DB2REORG\_LONGLOB
- No Table Truncation: DB2REORG\_NOTRUNCATE\_ONLINE
- Replace Compression Dictionary: DB2REORG\_RESET\_DICTIONARY
- Keep Compression Dictionary: DB2REORG\_KEEP\_DICTIONARY
- Reclaim Extents: DB2REORG\_RECLAIM\_EXTS

In addition to the preceding attribute settings, the following attributes are listed in the CLP output of the GET SNAPSHOT FOR TABLES command. These attribute settings are based on the values of other attribute settings or table reorganize monitor elements.

- Reclustering: If the value of the reorg\_index\_id monitor element is non-zero, then the table reorganize operation has this attribute.
- Reclaiming: If the value of the reorg\_index\_id monitor element is zero, then the table reorganize operation has this attribute.
- Inplace Table Reorg: If the reorg\_status monitor element has a value that is not null, then the in-place (online) reorganization method is in use.
- Table Reorg: If the reorg\_phase monitor element has a value that is not null, then the classic (offline) reorganization method is in use.
- Recluster Via Table Scan: If the DB2REORG\_INDEXSCAN flag is not set, then the table reorganize operation has this attribute.
- Reorg Data Only: If the DB2REORG\_LONGLOB flag is not set, then the table reorganize operation has this attribute.

## reorg\_xml\_regions\_compressed – XML regions compressed monitor element

Number of XML regions that were compressed during the table reorganization process.

Table 945. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

---

## reorg\_xml\_regions\_rejected\_for\_compression – XML regions rejected for compression monitor element

Number of XML regions that were not compressed during the table reorganization process.

Table 946. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table_reorg	Basic

---

## request\_exec\_time\_avg - Request execution time average monitor element

Arithmetic mean of the execution times for requests associated with this service subclass since the last reset. If the internally tracked average has overflowed, the value -2 is returned. This monitor element returns -1 when COLLECT AGGREGATE REQUEST DATA for the service subclass is set to NONE. Units are milliseconds.

When you remap activities between service subclasses with a REMAP ACTIVITY action, the request\_exec\_time\_avg mean counts the partial request in each subclass involved in remapping.

Table 947. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-

### Usage

Use this statistic to quickly understand the average amount of time that is spent processing each request on a database partition in this service subclass.

This average can also be used to determine whether or not the histogram template used for the request execution time histogram is appropriate. Compute the average request execution time from the request execution time histogram. Compare the computed average with this monitor element. If the computed average deviates from the true average reported by this monitor element, consider altering the histogram template for the request execution time histogram, using a set of bin values that are more appropriate for your data.

---

## rf\_log\_num - Log Being Rolled Forward

The log being processed.

**Element identifier**  
rf\_log\_num



**Element type**  
information

*Table 948. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	rollforward	Basic

**Usage** If a rollforward is in progress, this element identifies the log involved.

---

## rf\_status - Log Phase

The status of the recovery.

**Element identifier**  
rf\_status

**Element type**  
information

*Table 949. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	rollforward	Basic

**Usage** This element indicates the progression of a recovery. It indicates if the recovery is in an undo (rollback) or redo (rollforward) phase.

---

## rf\_timestamp - Rollforward Timestamp

The timestamp of the last committed transaction..

**Element identifier**  
rf\_timestamp

**Element type**  
timestamp

*Table 950. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	rollforward	Timestamp

**Usage** If a rollforward is in progress, this is the timestamp of the last committed transaction processed by rollforward recovery. This is an indicator of how far the rollforward operation has progressed.

---

## rf\_type - Rollforward Type

The type of rollforward in progress.

**Element identifier**  
rf\_type

**Element type**  
information

Table 951. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	rollforward	Basic

**Usage** An indicator of whether recovery is happening at a database or table space level.

## rollback\_sql\_stmts - Rollback Statements Attempted

The total number of SQL ROLLBACK statements that have been attempted.

**Element identifier**

rollback\_sql\_stmts

**Element type**

counter

Table 952. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Database	dbase_remote	Basic
Application	appl	Basic
Application	appl_remote	Basic
DCS Database	dcs_dbase	Basic
DCS Application	dcs_appl	Basic

For snapshot monitoring, this counter can be reset.

Table 953. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** A rollback can result from an application request, a deadlock, or an error situation. This element **only** counts the number of rollback statements issued from applications.

At the application level, this element can help you determine the level of database activity for the application and the amount of conflict with other applications. At the database level, it can help you determine the amount of activity in the database and the amount of conflict between applications on the database.

**Note:** You should try to minimize the number of rollbacks, since higher rollback activity results in lower throughput for the database.

It may also be used to calculate the total number of units of work, by calculating the sum of the following:

```

commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks

```

---

## rolled\_back\_agent\_id - Rolled Back Agent

Agent that was rolled back when a deadlock occurred.

**Element identifier**

rolled\_back\_agent\_id

**Element type**

information

*Table 954. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	event_deadlock	-

**Usage** A system administrator can use this information to determine which application did not complete its updates, and determine which applications should be restarted.

---

## rolled\_back\_appl\_id - Rolled Back Application

Application id that was rolled back when a deadlock occurred.

**Element identifier**

rolled\_back\_appl\_id

**Element type**

information

*Table 955. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	event_deadlock	-

**Usage** A system administrator can use this information to determine which application did not complete its updates, and determine which applications should be restarted.

---

## rolled\_back\_participant\_no - Rolled back application participant monitor element

The participant number identifying the rolled back application.

*Table 956. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks <sup>1</sup>	event_deadlock	-

**1** This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

**Usage**

A system administrator can use this information to determine which application did not complete its updates, and determine which application should be started.

---

## rolled\_back\_sequence\_no - Rolled Back Sequence Number

The sequence number of the application that was rolled back when a deadlock occurred.

### Element identifier

rolled\_back\_sequence\_no

### Element type

information

*Table 957. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	event_deadlock	-

**Usage** A system administrator can use this information to determine which application did not complete its updates, and determine which applications should be restarted.

---

## root\_node\_splits - Root node splits monitor element

Number of times the root node of the index was split during an insert operation.

*Table 958. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected

---

## routine\_id - Routine ID monitor element

This monitor element is a unique routine identifier. It returns zero if the activity is not part of a routine.

*Table 959. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### Usage

---

## rows\_deleted - Rows deleted monitor element

This is the number of row deletions attempted.

*Table 960. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLE table function - Get table metrics	Always collected

Table 961. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Database	dbase_remote	Basic
Application	appl	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

Table 962. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** You can use this element to gain insight into the current level of activity within the database.

This count does not include the attempts counted in the **int\_rows\_deleted** monitor element.

---

## rows\_fetched - Rows fetched monitor element

The number of rows read from the table.

This monitor element is an alias of the **rows\_read** monitor element.

**Note:** This monitor element reports only the values for the database partition for which this information is recorded. On DPF systems, these values may not reflect the correct totals for the whole activity.

### Element identifier

rows\_fetched

### Element type

counter

-->

Table 963. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	Statement

### Usage

See the **rows\_read** monitor element for details.

---

## rows\_inserted - Rows inserted monitor element

The number of row insertions attempted.

Table 964. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLE table function - Get table metrics	Always collected

Table 965. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Database	dbase_remote	Basic
Application	appl	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

Table 966. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** You can use this element to gain insight into the current level of activity within the database.

In a federated system, multiple rows can be inserted, per INSERT statement, because the federated server can push INSERT FROM SUBSELECT to the data source, when appropriate.

This count does not include the attempts counted in the `int_rows_inserted` monitor element.

## rows\_modified - Rows modified monitor element

The number of rows inserted, updated, or deleted.

This monitor element is an alias of the `rows_written` monitor element.

Table 967. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE

Table 967. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 968. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Activities	event_activity	Statement

## Usage

See the **rows\_written** monitor element for details.

## rows\_read - Rows read monitor element

The number of rows read from the table.

Table 969. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 969. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_TABLE table function - Get table metrics	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 970. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Table	table	Table
Application	appl	Basic
Application	stmt	Basic
Application	subsection	Statement
Dynamic SQL	dynsql	Statement

For snapshot monitoring, this counter can be reset.

Table 971. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Connection	event_conn	-
Tables	event_table	-
Statements	event_stmt	-
Transactions	event_xact	-



## Usage

This element helps you identify tables with heavy usage for which you may want to create additional indexes. To avoid the maintenance of unnecessary indexes, use the SQL EXPLAIN statement to determine if the package uses an index.

This count is *not* the number of rows that were returned to the calling application. Rather, it is the number of rows that had to be read in order to return the result set. For example, the following statement returns one row to the application, but many rows are read to determine the average salary:

```
SELECT AVG(SALARY) FROM USERID.EMPLOYEE
```

This count includes the value in the **overflow\_accesses** monitor element. Additionally, this count does not include any index accesses. That is, if an access plan uses index access only and the table is not touched to look at the actual row, then the value of the **rows\_read** monitor element is not incremented.

---

## rows\_returned - Rows returned monitor element

The number of rows that have been selected and returned to the application. This element has a value of 0 for partial activity records (for example, if an activity is collected while it is still executing or when a full activity record could not be written to the event monitor due to memory limitations).

This monitor element is an alias of the **fetch\_count** monitor element.

*Table 972. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE

Table 972. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 973. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Activities	event_activity	-

## Usage

This element can be used to help determine thresholds for rows returned to the application or can be used to verify that such a threshold is configured correctly and doing its job.

---

## rows\_returned\_top - Actual rows returned top monitor element

The high watermark for the actual rows returned of DML activities at all nesting levels in a service class or work class. For service classes, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service class is set to NONE. For work classes, this monitor element returns -1 if no COLLECT AGGREGATE ACTIVITY DATA work action is specified for the work class. For workloads, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the workload is set to NONE.

For service classes, when you remap activities between service subclasses with a REMAP ACTIVITY action, only the rows\_returned\_top high watermark of the service subclass where an activity completes is updated. High watermarks of service subclasses an activity is mapped to but does not complete in are unaffected.

Table 974. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wcstats	-
Statistics	event_wlstats	-

## Usage

Use this element to know the highest DML activity actual rows returned reached on a partition for a service class, workload, or work class in the time interval collected.

---

## rows\_selected - Rows Selected

This is the number of rows that have been selected and returned to the application.

*Table 975. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Database	dbase_remote	Basic
Application	appl	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

*Table 976. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** You can use this element to gain insight into the current level of activity within the database.

This element does not include a count of rows read for actions such as COUNT(\*) or joins.

For a federated system, you can compute the average time to return a row to the federated server from the data source:

$$\text{average time} = \text{rows returned} / \text{aggregate query response time}$$

You can use these results to modify CPU speed or communication speed parameters in SYSCAT.SERVERS. Modifying these parameters can impact whether the optimizer does or does not send requests to the data source.

**Note:** This element is collected at the dcs\_dbase and dcs\_appl snapshot monitor logical data groups if the gateway being monitored is at DB2 database version 7.2 or lower.

---

## rows\_updated - Rows updated monitor element

This is the number of row updates attempted.

*Table 977. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLE table function - Get table metrics	Always collected

Table 978. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Database	dbase_remote	Basic
Application	appl	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

Table 979. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** You can use this element to gain insight into the current level of activity within the database.

This value does not include updates counted in the **int\_rows\_updated** monitor element. However, rows that are updated by more than one update statement are counted for each update.

---

## rows\_written - Rows Written

This is the number of rows changed (inserted, deleted or updated) in the table.

Table 980. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table	Basic
Application	appl	Basic
Application	stmt	Basic
Application	subsection	Statement
Dynamic SQL	dynsql	Statement

For snapshot monitoring, this counter can be reset.

Table 981. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-
Tables	event_table	-
Statements	event_stmt	-
Transactions	event_xact	-

**Usage** A high value for table-level information indicates there is heavy usage of the table and you may want to use the Run Statistics (RUNSTATS) utility to maintain efficiency of the packages used for this table.

For application-connections and statements, this element includes the number of rows inserted, updated, and deleted in temporary tables.

At the application, transaction, and statement levels, this element can be useful for analyzing the relative activity levels, and for identifying candidates for tuning.

---

## rqsts\_completed\_total - Total requests completed monitor element

The total number of requests executed, including both application and internal requests. For service subclasses, this monitor element is only updated where the request completes. If the request moved between different service subclasses, it is not counted twice.

*Table 982. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

*Table 983. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

---

## sc\_work\_action\_set\_id - Service class work action set ID monitor element

If this activity has been categorized into a work class of service class scope, this monitor element displays the ID of the work action set associated with the work class set to which the work class belongs. Otherwise, this monitor element displays the value of 0.

Table 984. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
WLM_GET_ACTIVITY_DETAILS_COMPLETE table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 985. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

## Usage

This element can be used with the `sc_work_class_id` element to uniquely identify the service class work class of the activity, if one exists.

## sc\_work\_class\_id - Service class work class ID monitor element

If this activity has been categorized into a work class of service class scope, this monitor element displays the ID of the work class assigned to this activity. Otherwise, this monitor element displays the value of 0.

Table 986. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
WLM_GET_ACTIVITY_DETAILS_COMPLETE table function - Get complete activity details (reported in DETAILS XML document)	Always collected

Table 987. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

## Usage

This element can be used with the `sc_work_action_set_id` element to uniquely identify the service class work class of the activity, if one exists.

## sec\_log\_used\_top - Maximum Secondary Log Space Used

The maximum amount of secondary log space used (in bytes).

Table 988. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Table 989. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

**Usage** You may use this element in conjunction with `sec_logs_allocated` and

*tot\_log\_used\_top* to show your current dependency on secondary logs. If this value is high, you may need larger log files, or more primary log files, or more frequent COMMIT statements within your application.

As a result, you may need to adjust the following configuration parameters:

- logfilsiz
- logprimary
- logsecond
- logretain

The value will be zero if the database does not have any secondary log files. This would be the case if there were none defined.

**Note:** While the database system monitor information is given in bytes, the configuration parameters are set in pages, which are each 4K bytes.

## sec\_logs\_allocated - Secondary Logs Allocated Currently

The total number of secondary log files that are currently being used for the database.

Table 990. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

**Usage** You may use this element in conjunction with *sec\_log\_used\_top* and *tot\_log\_used\_top* to show your current dependency on secondary logs. If this value is consistently high, you may need larger log files, or more primary log files, or more frequent COMMIT statements within your application.

As a result, you may need to adjust the following configuration parameters:

- logfilsiz
- logprimary
- logsecond
- logretain

## section\_env - Section environment monitor element

A handle that gives details of an activity's section.

Table 991. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activitystmt	-

### Usage

This element is to be used with future IBM tools for extracting section information for the activity described in this record

---

## section\_number - Section number monitor element

The internal section number in the package for the static SQL statement currently processing or most recently processed.

Table 992. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected

Table 993. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement
DCS Statement	dcs_stmt	Statement

Table 994. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details <sup>1</sup>	event_detailed_dlconn	-
Statements	event_stmt	-
Activities	event_activitystmt	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

### Usage

For static SQL statements, you can use this element along with **creator**, **package\_version\_id**, and **package\_name** monitor elements to query the SYSCAT.STATEMENTS system catalog table and obtain the static SQL statement text, using the sample query as follows:

```
SELECT SEQNO, SUBSTR(TEXT,1,120)
FROM SYSCAT.STATEMENTS
WHERE PKGNAME = 'package_name' AND
      PKGSHEMA = 'creator' AND
      VERSION = 'package_version_id' AND
      SECTNO = section_number
ORDER BY SEQNO
```

**Note:** Exercise caution in obtaining static statement text, because this query against the system catalog table could cause lock contentions. Whenever possible, only use this query when there is little other activity against the database.

---

## section\_type - Section type indicator monitor element

Indicates whether the SQL statement section is dynamic or static.



Table 995. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

## Usage

The possible values for this monitor element are the following:

- D: dynamic
- S: static

---

## select\_sql\_stmts - Select SQL Statements Executed

The number of SQL SELECT statements that were executed.

Table 996. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Database	dbase_remote	Basic
Table Space	tablespace	Basic
Application	appl	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

Table 997. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** You can use this element to determine the level of database activity at the application or database level.

You can also use the following formula to determine the ratio of SELECT statements to the total statements:

$$\frac{\text{select\_sql\_stmts}}{(\text{static\_sql\_stmts} + \text{dynamic\_sql\_stmts})}$$

This information can be useful for analyzing application activity and throughput.

---

## select\_time - Query Response Time

This element contains the aggregate amount of time, in milliseconds, that it has taken this data source to respond to queries from all applications or a single application running on this federated server instance since the start of the federated server instance or the last reset of the database monitor counters, whichever is latest.

**Note:** Due to query blocking, not all attempts by the federated server to retrieve a row result in communication processing; the request to get the next row can potentially be satisfied from a block of returned rows. As a result, the aggregate query response time does not always indicate processing at the data source, but it usually indicates processing at either the data source or client.

*Table 998. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Timestamp
Application	appl_remote	Timestamp

For snapshot monitoring, this counter can be reset.

## Usage

Use this element to determine how much actual time is spent waiting for data from this data source. This can be useful in capacity planning and tuning the CPU speed and communication rates in SYSCAT.SERVERS. Modifying these parameters can impact whether the optimizer does or does not send requests to the data source.

The response time is measured as the difference in time between the time the federated server requests a row from the data source, and the time the row is available for the federated server to use.

---

## sequence\_no - Sequence number monitor element

This identifier is incremented whenever a unit of work ends (that is, when a COMMIT or ROLLBACK terminates a unit of work). Together, the **appl\_id** and **sequence\_no** uniquely identify a transaction.

*Table 999. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_id_info	Basic
DCS Application	dcs_appl_info	Basic

*Table 1000. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-
Connections	event_connheader	-
Statements	event_stmt	-
Transactions	event_xact	-
Deadlocks	event_dlconn	-
Deadlocks with Details	event_detailed_dlconn	-
Deadlocks with Details History	event_detailed_dlconn	-
Deadlocks with Details History	event_stmt_history	-
Deadlocks with Details History Values	event_detailed_dlconn	-

Table 1000. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks with Details History Values	event_stmt_history	-

## sequence\_no\_holding\_lk - Sequence Number Holding Lock

The sequence number of the application that is holding a lock on the object that this application is waiting to obtain.

**Element identifier**

sequence\_no\_holding\_lk

**Element type**

information

Table 1001. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic
Lock	appl_lock_list	Basic

Table 1002. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	event_dlconn	-
Deadlocks with Details	event_detailed_dlconn	-

**Usage** This identifier is used in tandem with `appl_id` to uniquely identify a transaction that is holding a lock on the object that this application is waiting to obtain.

## server\_db2\_type - Database Manager Type at Monitored (Server) Node

Identifies the type of database manager being monitored.

Table 1003. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	collected	Basic

**Usage** It contains one of the following types of configurations for the database manager:

**API Symbolic Constant**

**Command Line Processor Output**

**sqlf\_nt\_server**

Database server with local and remote clients

**sqlf\_nt\_stand\_req**

Database server with local clients

The API symbolic constants are defined in the include file `sqlutil.h`.

---

## server\_instance\_name - Server Instance Name

The name of the database manager instance for which the snapshot was taken.

**Element identifier**

server\_instance\_name

**Element type**

information

*Table 1004. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	collected	Basic

*Table 1005. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Event Log Header	event_log_header	-

**Usage** If more than one instance of the database manager is present on the same system, this data item is used to uniquely identify the instance for which the snapshot call was issued. This information can be useful if you are saving your monitor output in a file or database for later analysis, and you need to differentiate the data from different instances of the database manager.

---

## server\_platform - Server Operating System

The operating system running the database server.

**Element identifier**

server\_platform

**Element type**

information

*Table 1006. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

*Table 1007. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

**Usage** This element can be used for problem determination for remote applications. Values for this field can be found in the header file *sqlmon.h*.

---

## server\_prdid - Server Product/Version ID

The product and version that is running on the server.

*Table 1008. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	collected	Basic

Table 1009. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Event Log Header	event_log_header	-

**Usage** It is in the form PPPVVRRM, where:

- PPP** is SQL
- VV** identifies a 2-digit version number (with high-order 0 in the case of a 1-digit version)
- RR** identifies a 2-digit release number (with high-order 0 in the case of a 1-digit release)
- M** identifies a 1-character modification level (0-9 or A-Z)

---

## server\_version - Server Version

The version of the server returning the information.

Table 1010. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	collected	Basic

### Usage

This field identifies the level of the database server collecting database system monitor information. This allows applications to interpret the data based on the level of the server returning the data. Valid values are:

**SQLM\_DBMON\_VERSION1**

Data was returned by DB2 Version 1

**SQLM\_DBMON\_VERSION2**

Data was returned by DB2 Version 2

**SQLM\_DBMON\_VERSION5**

Data was returned by DB2<sup>®</sup> Universal Database<sup>™</sup> Version 5

**SQLM\_DBMON\_VERSION5\_2**

Data was returned by DB2 Universal Database Version 5.2

**SQLM\_DBMON\_VERSION6**

Data was returned by DB2 Universal Database Version 6

**SQLM\_DBMON\_VERSION7**

Data was returned by DB2 Universal Database Version 7

**SQLM\_DBMON\_VERSION8**

Data was returned by DB2 Universal Database Version 8

**SQLM\_DBMON\_VERSION9**

Data was returned by DB2 Database for Linux, UNIX, and Windows Version 9

**SQLM\_DBMON\_VERSION9\_5**

Data was returned by DB2 Database for Linux, UNIX, and Windows Version 9.5

---

## service\_class\_id - Service class ID monitor element

Unique ID of service subclass. For a workload, this ID represents the service subclass that the workload is mapped to. For a unit of work, this ID represents the service subclass ID of the workload that the connection issuing the unit of work is associated with.

*Table 1011. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	Always collected
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics	Always collected
MON_GET_WORKLOAD table function - Get workload metrics	Always collected
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

*Table 1012. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Locking	-	-
Unit of work	-	-
Statistics	event_histogrambin	-
Statistics	event_scstats	-

### Usage

The value of this element matches a value from column SERVICECLASSID of view SYSCAT.SERVICECLASSES. Use this element to look up the service subclass name, or link information about a service subclass from different sources. For example, join service class statistics with histogram bin records.

---

## service\_level - Service Level

This is the current corrective service level of the DB2 instance.

*Table 1013. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

---

## service\_subclass\_name - Service subclass name monitor element

The name of a service subclass.

*Table 1014. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	Always collected
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected

*Table 1015. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Locking	-	-
Unit of work	-	-
Activities	event_activity	-
Statistics	event_scstats	-
Statistics	event_qstats	-

### Usage

Use this element in conjunction with other activity elements for analysis of the behavior of an activity or with other statistics elements for analysis of a service class or threshold queue.

---

## service\_superclass\_name - Service superclass name monitor element

The name of a service superclass.

*Table 1016. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	Always collected
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	Always collected
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected

Table 1016. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected

Table 1017. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	-	-
Activities	event_activity	-
Statistics	event_scstats	-
Statistics	event_qstats	-

## Usage

Use this element in conjunction with other activity elements for analysis of the behavior of an activity or with other statistics elements for analysis of a service class or threshold queue.

---

## session\_auth\_id - Session authorization ID monitor element

The current authorization ID for the session being used by this application. For monitoring workload management activities, this monitor element describes the session authorization ID under which the activity was injected into the system.

Table 1018. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
Lock	appl_lock_list	Basic

Table 1019. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-
Activities	event_activity	-
Threshold violations	event_activity	-

## Usage

You can use this element to determine what authorization ID is being used to prepare SQL statements, execute SQL statements, or both. This monitor element does not report any session authorization ID values set within executing stored procedures.



---

## shr\_workspace\_num\_overflows - Shared Workspace Overflows

The number of times that shared workspaces overflowed the bounds of their allocated memory.

**Note:** This monitor element has been deprecated. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

*Table 1020. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

*Table 1021. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** Use this element with shr\_workspace\_size\_top to determine whether the size of the Shared Workspaces need to be increased to avoid overflowing. Overflows of Shared Workspaces may cause performance degradation as well as out of memory errors from the other heaps allocated out of application shared memory.

At the database level, the element reported will be from the same shared workspace as that which was reported as having the Maximum Shared Workspace Size. At the application level, it is the number of overflows for the workspace used by the current application.

---

## shr\_workspace\_section\_inserts - Shared Workspace Section Inserts

Number of inserts of SQL sections by applications into shared workspaces.

**Note:** This monitor element has been deprecated. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

*Table 1022. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

*Table 1023. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** The working copy of executable sections are stored in shared workspaces. This counter indicates when a copy was not available and had to be inserted.

At the database level, it is the cumulative total of all inserts for every application across all shared workspaces in the database. At the application level, it is the cumulative total of all inserts for all sections in the shared workspace for this application.

---

## shr\_workspace\_section\_lookups - Shared Workspace Section Lookups

Lookups of SQL sections by applications in shared workspaces.

**Note:** This monitor element has been deprecated. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

*Table 1024. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

*Table 1025. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** Each application has access to a shared workspace where the working copy of executable sections are kept.

This counter indicates how many times shared workspaces were accessed in order to locate a specific section for an application. At the database level, it is the cumulative total of all lookups for every application across all Shared Workspaces in the database. At the application level, it is the cumulative total of all lookups for all sections in the shared workspace for this application.

You can use this element in conjunction with Shared Workspace Section Inserts to tune the size of shared workspaces. The size of the shared workspace is controlled by the `app_ctl_heap_sz` configuration parameter.

---

## shr\_workspace\_size\_top - Maximum Shared Workspace Size

The largest size reached by shared workspaces.

**Note:** This monitor element has been deprecated. Using this monitor element will not generate an error. However, it does not return a valid value. This monitor element is no longer recommended and might be removed in a future release.

*Table 1026. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Table 1026. Snapshot Monitoring Information (continued)

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

Table 1027. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** This element indicates the maximum number of bytes the shared workspaces required for the workload run against the database since it was activated. At the database level, it is the maximum size reached by all of the shared workspaces. At the application level, it is the maximum size of the shared workspace used by the current application.

If a shared workspace overflowed, then this element contains the largest size reached by that shared workspace during the overflow. Check Shared Workspace Overflows to determine if such a condition occurred.

When the shared workspace overflows, memory is temporarily borrowed from other entities in application shared memory. This can result in memory shortage errors from these entities or possibly performance degradation. You can reduce the chance of overflow by increasing APP\_CTL\_HEAP\_SZ.

---

## smallest\_log\_avail\_node - Node with Least Available Log Space

This element is only returned for global snapshots and indicates the node with the least amount (in bytes) of available log space.

**Element identifier**  
smallest\_log\_avail\_node

**Element type**  
information

Table 1028. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

**Usage** Use this element, in conjunction with appl\_id\_oldest\_xact, to ensure that adequate log space is available for the database. In a global snapshot, appl\_id\_oldest\_xact, total\_log\_used, and total\_log\_available correspond to the values on this node.

---

## sort\_heap\_allocated - Total Sort Heap Allocated

The total number of allocated pages of sort heap space for all sorts at the level chosen and at the time the snapshot was taken.

Table 1029. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic
Database	dbase	Basic

**Usage** The amount of memory allocated for each sort may be some or all of the available sort heap size. Sort heap size is the amount of memory available for each sort as defined in the *sortheap* database configuration parameter.

It is possible for a single application to have concurrent sorts active. For example, in some cases a SELECT statement with a subquery can cause concurrent sorts.

Information may be collected at two levels:

- At the database manager level, it represents the sum of sort heap space allocated for all sorts in all active databases in the database manager
- At the database level, it represents the sum of the sort heap space allocated for all sorts in a database.

Normal memory estimates do not include sort heap space. If excessive sorting is occurring, the extra memory used for the sort heap should be added to the base memory requirements for running the database manager. Generally, the larger the sort heap, the more efficient the sort. Appropriate use of indexes can reduce the amount of sorting required.

You may use the information returned at the database manager level to help you tune the *sheapthres* configuration parameter. If the element value is greater than or equal to *sheapthres*, it means that the sorts are not getting the full sort heap as defined by the *sortheap* parameter.

---

## sort\_heap\_top - Sort private heap high watermark

The private sort memory high watermark, in 4 KB pages, across the database manager.

Table 1030. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	db2	Basic

**Usage** This element can be used to determine if the SHEAPTHRES configuration parameter is set to an optimal value. For example, if this watermark approaches or exceeds SHEAPTHRES, it is likely that SHEAPTHRES should be increased. This is because private sorts are given less memory whenever SHEAPTHRES is exceeded, and this can adversely affect system performance.

---

## sort\_overflows - Sort overflows monitor element

The total number of sorts that ran out of sort heap and may have required disk space for temporary storage.

Table 1031. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1031. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 1032. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic
Application	stmt	Basic
Dynamic SQL	dynsql	Basic

For snapshot monitoring, this counter can be reset.

Table 1033. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Connection	event_conn	-
Statements	event_stmt	-

Table 1033. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	Statement, Sort

## Usage

At a database or application level, use this element in conjunction with **total\_sorts** to calculate the percentage of sorts that had to overflow to disk. If this percentage is high, you may want adjust the database configuration by increasing the value of **sortheap**.

At a statement level, use this element to identify statements that require large sorts. These statements may benefit from additional tuning to reduce the amount of sorting required.

When a sort overflows, additional overhead will be incurred because the sort will require a merge phase and can potentially require more I/O, if data needs to be written to disk.

This element provides information for one statement, one application, or all applications accessing one database.

---

## sort\_shrheap\_allocated - Sort Share Heap Currently Allocated

Total amount of shared sort memory allocated in the database.

Table 1034. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

**Usage** This element can be used to assess the threshold for shared sort memory. If this value is frequently much higher or lower than the current shared sort memory threshold, it is likely that the threshold should be adjusted.

**Note:** The "shared sort memory threshold" is determined by the value of the SHEAPTHRES database manager configuration parameter if the SHEAPTHRES\_SHR database configuration parameter is 0. Otherwise, it is determined by the value of SHEAPTHRES\_SHR.

---

## sort\_shrheap\_top - Sort share heap high watermark

Database-wide shared sort memory high watermark in 4 KB pages.

Table 1035. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

**Usage** This element can be used to assess whether or not SHEAPTHRES (or SHEAPTHRES\_SHR) is set to an optimal value. For example, if this high watermark is persistently much lower than the shared sort memory threshold, it is likely that this threshold needs to be decreased, thus freeing memory for other database functions. Conversely, if this high watermark begins to approach the shared sort memory threshold, then this might

indicate that this threshold needs to be increased. This is important because the shared sort memory threshold is a hard limit. When the total amount of sort memory reaches this threshold, no more shared sorts can be initiated.

This element, along with the high watermark for private sort memory, can also help users determine if the threshold for shared and private sorts need to be set independently of each other. Normally, if the SHEAPTHRES\_SHR database configuration option has a value of 0, then the shared sort memory threshold is determined by the value of the SHEAPTHRES database manager configuration option. However, if there is a large discrepancy between the private and shared sort memory high watermarks, this might be an indication that the user needs to override SHEAPTHRES and set SHEAPTHRES\_SHR to a more appropriate value that is based on the shared sort memory high watermark.

---

## source\_service\_class\_id - Source service class ID monitor element

The ID of the service subclass from which an activity was remapped when the threshold violation record to which this element belongs was generated. This element has a value of zero when the threshold action is anything other than a REMAP ACTIVITY action.

### Element identifier

source\_service\_class\_id

### Element type

Information

*Table 1036. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Threshold violations	event_thresholdviolations	-

## Usage

Use this element to trace the path of an activity through the service classes to which it was remapped. It can also be used to compute aggregates of how many activities were mapped out of a given service subclass.

---

## sp\_rows\_selected - Rows Returned by Stored Procedures

This element contains the number of rows sent from the data source to the federated server at the start of the federated server instance, or the last reset of the database monitor counters as a result of stored procedure operations for this application.

*Table 1037. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

**Usage** This element has several uses. You can use it to compute the average

number of rows sent to the federated server from the data source, per stored procedure, with the following formula:

$$\begin{aligned} & \text{rows per stored procedure} \\ &= \text{rows returned} \\ & / \# \text{ of stored procedures invoked} \end{aligned}$$

You can also compute the average time to return a row to the federated server from the data source for this application:

$$\text{average time} = \text{aggregate stored proc. response time} / \text{rows returned}$$

---

## sql\_chains - Number of SQL Chains Attempted

Represents the number of SQL statements taking  $n$  data transmissions between the DB2 Connect gateway and the host during statement processing. The range  $n$  is specified by the *num\_transmissions\_group* element.

Table 1038. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Data Transmission	stmt_transmissions	Basic

For snapshot monitoring, this counter can be reset.

For example, if chaining is on, and if PREP and OPEN statements are chained together and the chain takes a total of two transmissions, *sql\_chains* is reported as "1" and *sql\_stmts* is reported as "2".

If chaining is off, then the *sql\_chains* count equals the *sql\_stmts* count.

**Usage** Use this element to get statistics on how many statements used 2, 3, 4 (and so on) data transmissions during their processing. (At least two data transmissions are necessary to process a statement: a send and a receive.) These statistics can give you a better idea of the database or application activity and network traffic at the database or application levels.

**Note:** The *sql\_stmts* monitor element represents the number of attempts made to send an SQL statement to the server. At the transmission level, all statements within the same cursor count as a single SQL statement.

---

## sql\_req\_id - Request Identifier for SQL Statement

The request identifier for an operation in an SQL statement.

Table 1039. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_stmt	-

**Usage** This identifier increments with each successive SQL operation processed by the database manager since the first application has connected to the database. Its value is unique across the database and uniquely identifies a statement operation.

---

## sql\_reqs\_since\_commit - SQL Requests Since Last Commit

Number of SQL requests that have been submitted since the last commit.



Table 1040. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Basic

**Usage** You can use this element to monitor the progress of a transaction.

---

## sql\_stmts - Number of SQL Statements Attempted

For data transmission snapshots, this element represents the number of SQL statements taking *n* data transmissions between the DB2 Connect gateway and the host during statement processing. The range *n* is specified by the *num\_transmissions\_group* element.

Table 1041. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Database	dcs_dbase	Basic
DCS Application	dcs_appl	Basic
Data Transmission	stmt_transmissions	Basic

For snapshot monitoring, this counter can be reset.

For DCS DATABASE snapshots, this statement count is the number of statements since the database was activated.

For DCS APPLICATION snapshots, this statement count is the number of statements since the connection to the database was established by this application.

**Usage** Use this element to measure database activity at the database or application level. To calculate the SQL statement throughput for a given period, you can divide this element by the elapsed time between two snapshots.

For the data transmission level: Use this element to get statistics on how many statements used 2, 3, 4 (and so on) data transmissions during their processing. (At least 2 data transmissions are necessary to process a statement: a send and a receive.) These statistics can give you a better idea of the database or application activity and network traffic at the database or application levels.

**Note:**

1. The *sql\_stmts* monitor element represents the number of attempts made to send an SQL statement to the server:
  - At the application level and database level, each SQL statement within a cursor is counted separately.
  - At the transmission level, all statements within the same cursor count as a single SQL statement.

---

## sqlca - SQL Communications Area (SQLCA)

The SQLCA data structure that was returned to the application at statement completion.

Table 1042. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_stmt	-
Activities	event_activity	-

## Usage

The SQLCA data structure can be used to determine if the statement completed successfully. For information about the content of the SQLCA, see “SQLCA (SQL communications area)” in *SQL Reference, Volume 1* or “SQLCA data structure” in *Administrative API Reference*.

---

## sqlrowsread\_threshold\_id - SQL rows read threshold ID monitor element

The ID of the SQLROWSREAD threshold that was applied to the activity.

Table 1043. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

Use this element to understand which SQLROWSREAD threshold, if any, was applied to the activity.

---

## sqlrowsread\_threshold\_value - SQL rows read threshold value monitor element

The upper bound of the SQLROWSREAD threshold that was applied to the activity.

Table 1044. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

Use this element to understand the value of the SQLROWSREAD threshold applied to the activity, if any.

---

## sqlrowsread\_threshold\_violated - SQL rows read threshold violated monitor element

This monitor element returns ‘Yes’ to indicate that the activity violated the SQLROWSREAD threshold. ‘No’ indicates that the activity has not yet violated the threshold.

Table 1045. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

Use this element to determine if the activity violated the SQLROWSREAD threshold that was applied to the activity.

---

## sqlrowsreadinsc\_threshold\_id - SQL rows read in service class threshold ID monitor element

The ID of the SQLROWSREADINSC threshold that was applied to the activity.

Table 1046. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

Use this element to understand which SQLROWSREADINSC threshold, if any, was applied to the activity.

---

## sqlrowsreadinsc\_threshold\_value - SQL rows read in service class threshold value monitor element

The upper bound of the SQLROWSREADINSC threshold that was applied to the activity.

Table 1047. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

Use this element to understand the value of the SQLROWSREADINSC threshold applied to the activity, if any.

---

## sqlrowsreadinsc\_threshold\_violated - SQL rows read in service class threshold violated monitor element

This monitor element returns 'Yes' to indicate that the activity violated the SQLROWSREADINSC threshold. 'No' indicates that the activity has not yet violated the threshold.

Table 1048. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### Usage

Use this element to determine if the activity violated the SQLROWSREADINSC threshold that was applied to the activity.

---

## sqlrowsreturned\_threshold\_id - SQL rows read returned threshold ID monitor element

The ID of the SQLROWSRETURNED threshold that was applied to the activity.

Table 1049. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### Usage

Use this element to understand which SQLROWSRETURNED threshold, if any, was applied to the activity.

---

## sqlrowsreturned\_threshold\_value - SQL rows read returned threshold value monitor element

The upper bound of the SQLROWSRETURNED threshold that was applied to the activity.

Table 1050. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### Usage

Use this element to understand the value of the SQLROWSRETURNED threshold applied to the activity, if any.

---

## sqlrowsreturned\_threshold\_violated - SQL rows read returned threshold violated monitor element

This monitor element returns 'Yes' to indicate that the activity violated the SQLROWSRETURNED threshold. 'No' indicates that the activity has not yet violated the threshold.

Table 1051. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### Usage

Use this element to determine if the activity violated the SQLROWSRETURNED threshold that was applied to the activity.

---

## sqltempespace\_threshold\_id - SQL temporary space threshold ID monitor element

The ID of the SQLTEMPSPACE threshold that was applied to the activity.

Table 1052. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### Usage

Use this element to understand which SQLTEMPSPACE threshold, if any, was applied to the activity.

---

## sqltempespace\_threshold\_value - SQL temporary space threshold value monitor element

The upper bound of the SQLTEMPSPACE threshold that was applied to the activity.

Table 1053. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

### Usage

Use this element to understand the value of the SQLTEMPSPACE threshold applied to the activity, if any.

---

## sqltempespace\_threshold\_violated - SQL temporary space threshold violated monitor element

This monitor element returns 'Yes' to indicate that the activity violated the SQLTEMPSPACE threshold. 'No' indicates that the activity has not yet violated the threshold.

Table 1054. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

## Usage

Use this element to determine if the activity violated the SQLTEMPSPACE threshold that was applied to the activity.

---

## ss\_exec\_time - Subsection Execution Elapsed Time

The time in seconds that it took a subsection to execute.

Table 1055. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

Table 1056. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_subsection	-

**Usage** Allows you to track the progress of a subsection.

---

## ss\_node\_number - Subsection Node Number

Node where the subsection was executed.

Table 1057. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

Table 1058. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_subsection	-

**Usage** Use to correlate each subsection with the database partition where it was executed.

---

## ss\_number - Subsection Number

Identifies the subsection associated with the returned information.

Table 1059. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

Table 1060. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_subsection	-

**Usage** This number relates to the subsection number in the access plan that can be obtained with db2expln.

---

## ss\_status - Subsection Status

The current status of an executing subsection.

Table 1061. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

**Usage** The current status values can be:

- executing (SQLM\_SSEXEC in sqlmon.h)
- waiting for a lock
- waiting to receive data on a table queue
- waiting to send data on a table queue

---

## ss\_sys\_cpu\_time - System CPU Time used by Subsection

The total system CPU time (in seconds and microseconds) used by the currently executing statement subsection.

**Element identifier**

ss\_sys\_cpu\_time

**Element type**

time

Table 1062. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Timestamp

Table 1063. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_subsection	Timestamp

**Usage** This element along with the other related CPU-time elements can help you understand the level of activity within an application, and can help you identify applications that could benefit from additional tuning.

System CPU represents the time spent in system calls. User CPU represents time spent executing database manager code.

---

## ss\_usr\_cpu\_time - User CPU Time used by Subsection

The total user CPU time (in seconds and microseconds) used by the currently executing statement subsection.

**Element identifier**  
ss\_usr\_cpu\_time

**Element type**  
time

*Table 1064. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Timestamp

*Table 1065. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_subsection	Timestamp

**Usage** This element along with the other related CPU-time elements can help you understand the level of activity within an application, and can help you identify applications that could benefit from additional tuning.

System CPU represents the time spent in system calls. User CPU represents time spent executing database manager code.

---

## start\_time - Event Start Time

The date and time of unit of work start, statement start, or deadlock detection. This element, in the event\_start API structure indicates the start of the event monitor.

*Table 1066. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_start	Timestamp
Transactions	event_xact	Timestamp
Statements	event_stmt	Timestamp
Deadlocks	event_deadlock	Timestamp
Deadlocks	event_dlconn	Timestamp
Deadlocks with Details	event_detailed_dlconn	Timestamp

**Usage** You can use this element to correlate the deadlock connection records to the deadlock event record, and in conjunction with *stop\_time* to calculate the elapsed statement or transaction execution time.

**Note:** When the Timestamp switch is OFF, this element reports "0".

---

## static\_sql\_stmts - Static SQL Statements Attempted

The number of static SQL statements that were attempted.

*Table 1067. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.



Table 1068. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** You can use this element to calculate the total number of successful SQL statements at the database or application level:

```
dynamic_sql_stmts
+ static_sql_stmts
- failed_sql_stmts
= throughput during monitoring period
```

---

## statistics\_timestamp - Statistics timestamp monitor element

The time at which this statistics record was generated.

Table 1069. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wlstats	-
Statistics	event_wcstats	-
Statistics	event_qstats	-
Statistics	event_histogrambin	-

### Usage

Use this element to determine when this statistics record was generated.

Use this element along with the **last\_wlm\_reset** element to identify the time interval over which the statistics in this statistics record were generated.

This monitor element can also be used to group together all statistics records that were generated for the same collection interval.

---

## stats\_cache\_size – Size of statistics cache monitor element

The current size of the statistics cache, which is used in a catalog partition to cache statistics information generated by real-time statistics gathering.

**Note:** Since the statistics cache resides in the catalog partition, only the snapshot taken at the catalog partition will report the statistics cache size. Snapshots taken at other partitions will report the value of zero instead. When taking a global snapshot, the values reported by all the database partitions are aggregated together.

Table 1070. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	-

Table 1071. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

## Usage

Use this element to determine the size of the current statistics cache. This value changes frequently. In order to evaluate system usage, take the snapshot at specific intervals over an extended period of time. Use this element to adjust the value of the `catalogcache_sz` configuration parameter.

---

## stats\_fabricate\_time – Total time spent on statistics fabrication activities monitor element

The total time spent on statistics fabrications by real-time statistics gathering, in milliseconds. Statistics fabrication is the statistics collection activity needed to generate statistics during query compilation. If this monitor element is collected at the database level, it represents the total time spent on real-time statistics gathering activities for all the applications running on the database. If it is collected at the statement level, it represents the time spent on the latest real-time statistics gathering activities for the statement. The times reported by all the database partitions are aggregated together.

Table 1072. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Statement
Dynamic SQL	dynsql	Statement

For snapshot monitoring, this element can be reset.

Table 1073. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Statement	event_stmt	-

## Usage

Use this element along with `stats_fabrications` to evaluate the performance impact of real-time statistics gathering at the database level. For snapshot monitor for dynamic SQL, you can use this element along with `total_exec_time` and `num_executions` to evaluate the impact of statistics fabrications. For the statement event monitor, you can combine this element with `stmt_start` and `stmt_stop` for further evaluation of real-time statistics gathering impact.

---

## stats\_fabrications – Total number of statistics fabrications monitor elements

The total number of statistics fabrications performed by real-time statistics during query compilation for all the database applications. Rather than obtaining statistics by scanning data stored in a table or an index, statistics are fabricated based on metadata maintained by the index and data manager. Values reported by all the database partitions are aggregated together.

Table 1074. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Statement

For snapshot monitoring, this counter can be reset.

Table 1075. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

### Usage

Use this element to determine the frequency of statistics fabrications in the database. This value changes frequently. In order to get a better overview of the system usage, take the snapshot at specific intervals over an extended period of time. When used in conjunction with **stats\_fabricate\_time**, this element can help you evaluate the impact of statistics fabrications.

---

## status\_change\_time - Application Status Change Time

The date and time the application entered its current status.

### Element identifier

status\_change\_time

### Element type

timestamp

Table 1076. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_id_info	Unit of Work, Timestamp
Lock	appl_lock_list	Unit of Work, Timestamp
DCS Application	dcs_appl_info	Unit of Work, Timestamp

**Usage** This element allows you to determine how long an application has been in its current status. If it has been in the same status for a long period of time, this may indicate that it has a problem.

---

## stmt\_elapsed\_time - Most Recent Statement Elapsed Time

The elapsed execution time of the most recently completed statement.

Table 1077. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement, Timestamp
DCS Statement	dcs_stmt	Statement, Timestamp

**Usage** Use this element as an indicator of the time it takes for a statement to complete.

---

## stmt\_first\_use\_time - Statement first use time

This element shows the first time the statement entry was processed. For cursor operations, **stmt\_first\_use\_time** shows when the cursor was opened. At application coordination nodes, this value reflects the application requests; at non-coordinator nodes, this value reflects when requests were received from the originating node.

Table 1078. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks with Details History Values	event_stmt_history	timestamp
Deadlocks with Details History	event_stmt_history	timestamp
Activities	event_activitystmt	timestamp

**Usage** Use this element in conjunction with other statement history entries to see the sequence of SQL statements that caused the deadlock.

---

## stmt\_history\_id - Statement history identifier

This numerical element shows the position in which the statement was run within the unit of work indicated by the `sequence_no` element, relative to other statement history elements. The earliest statement run in the unit of work will have the lowest value. If the same statement is run twice in the same unit of work, two different occurrences of the statement will be shown with two different `stmt_history_id` values.

Table 1079. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks with Details History Values	event_stmt_history	-
Deadlocks with Details History Values	event_data_value	-
Deadlocks with Details History	event_stmt_history	-

**Usage** You can use this information to see the sequence of SQL statements that caused the deadlock.

---

## inact\_stmthist\_sz - Statement history list size

When a detailed deadlock event monitor with history is running, this element reports the number of bytes being used from the database monitor heap (`MON_HEAP_SZ`) to keep track of the statement history list entries.

Table 1080. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
application	appl	-
database	db	-

**Usage** You can use this element when tuning the database monitor heap.

---

## stmt\_invocation\_id - Statement invocation identifier monitor element

This element shows the identifier (ID) of the routine invocation in which the SQL statement was run. The value indicates the number of routine invocations at the current nesting level that occurred while that level was active in the application.

Table 1081. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values <sup>1</sup>	event_stmt_history	-
Deadlocks with Details History <sup>1</sup>	event_stmt_history	-

- 1** This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

### Usage

You can use this element, along with **stmt\_nest\_level** monitor element, to uniquely identify an invocation of a particular SQL statement. You can also use this element in conjunction with other statement history entries to see the sequence of SQL statements that caused the deadlock.

---

## stmt\_isolation - Statement isolation

This element shows the isolation value in effect for the statement while it was being run.

Table 1082. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks with Details History Values	event_stmt_history	-
Deadlocks with Details History	event_stmt_history	-
Activities	event_activitystmt	-

The possible isolation level values are:

- SQLM\_ISOLATION\_LEVEL\_NONE 0 (no isolation level specified)
- SQLM\_ISOLATION\_LEVEL\_UR 1 (uncommitted read)
- SQLM\_ISOLATION\_LEVEL\_CS 2 (cursor stability)
- SQLM\_ISOLATION\_LEVEL\_RS 3 (read stability)

- `SQLM_ISOLATION_LEVEL_RR 4` (repeatable read)

**Usage** You can use this element in conjunction with other statement history entries to understand the cause of the deadlock and the execution behavior of a particular SQL statement.

## stmt\_last\_use\_time - Statement last use time monitor element

This element shows the last time the statement entry was processed. For cursor operations, `stmt_last_use_time` shows the time of the last action on the cursor where that action could be an open, fetch, or close. At application coordination nodes, this value reflects the application requests; at non-coordinator nodes, this value reflects when requests were received from the originating node.

*Table 1083. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks with Details History Values	event_stmt_history	timestamp
Deadlocks with Details History	event_stmt_history	timestamp
Activities	event_activitystmt	-

**Usage** Use this element in conjunction with other statement history entries to see the sequence of SQL statements that caused the deadlock.

## stmt\_lock\_timeout - Statement lock timeout monitor element

This element shows the lock timeout value in effect for the statement while it was being run.

*Table 1084. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values <sup>1</sup>	event_stmt_history	-
Deadlocks with Details History <sup>1</sup>	event_stmt_history	-
Activities	event_activitystmt	-

- 1** This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the `CREATE EVENT MONITOR FOR LOCKING` statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

### Usage

You can use this element in conjunction with other statement history entries to understand the cause of the deadlock and the execution behavior of a particular SQL statement.

---

## stmt\_nest\_level - Statement nesting level monitor element

This element shows the level of nesting or recursion in effect when the statement was being run; each level of nesting corresponds to nested or recursive invocation of a stored procedure or user-defined function (UDF).

Table 1085. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values <sup>1</sup>	event_stmt_history	-
Deadlocks with Details History <sup>1</sup>	event_stmt_history	-
Activities	event_activitystmt	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

### Usage

You can use this element, along with `stmt_invocation_id` monitor element, to uniquely identify an invocation of a particular SQL statement. You can also use this element in conjunction with other statement history entries to see the sequence of SQL statements that caused the deadlock.

---

## stmt\_node\_number - Statement Node

Node where the statement was executed.

Table 1086. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement

**Usage** Used to correlate each statement with the node where it was executed.

---

## stmt\_operation/operation - Statement operation monitor element

The statement operation currently being processed or most recently processed (if none currently running).

Table 1087. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement
DCS Statement	dcs_stmt	Statement

Table 1088. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details <sup>1</sup>	event_detailed_dlconn	-

Table 1088. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_stmt	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

## Usage

You can use this element to determine the operation that is executing or recently finished.

It can be one of the following.

For SQL operations:

- SELECT
- PREPARE
- EXECUTE
- EXECUTE IMMEDIATE
- OPEN
- FETCH
- CLOSE
- DESCRIBE
- STATIC COMMIT
- STATIC ROLLBACK
- FREE LOCATOR
- PREP\_COMMIT
- CALL
- PREP\_OPEN
- PREP\_EXEC
- COMPILE
- DROP PACKAGE

For non-SQL operations:

- RUN STATISTICS
- REORG
- REBIND
- REDISTRIBUTE
- GET TABLE AUTHORIZATION
- GET ADMINISTRATIVE AUTHORIZATION

**Note:** API users should refer to the `sqlmon.h` header file containing definitions of database system monitor constants.



---

## stmt\_pkgcache\_id - Statement package cache identifier

This element shows the internal package cache identifier (ID) for a dynamic SQL statement.

*Table 1089. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected

*Table 1090. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Dynamic SQL	dynsql	Basic

*Table 1091. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values <sup>1</sup>	event_stmt_history	-
Deadlocks with Details History <sup>1</sup>	event_stmt_history	-
Activities	event_activitystmt	-

- 1** This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

### Usage

In a multi-partitioned environment, each partition has a unique statement ID for a cached statement. A given statement may not have the same ID across partitions.

In a global dynamic SQL snapshot, only the first statement ID is returned.

---

## stmt\_query\_id - Statement query identifier monitor element

This element shows the internal query identifier (ID) given to any SQL statement used as a cursor.

*Table 1092. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values <sup>1</sup>	event_stmt_history	-
Deadlocks with Details History <sup>1</sup>	event_stmt_history	-

Table 1092. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activitystmt	-

## Usage

You can use this element, along with the **stmt\_nest\_level** monitor element, to uniquely identify an invocation of a particular SQL statement. You can also use this element in conjunction with other statement history entries to understand the cause of the deadlock.

---

## stmt\_sorts - Statement Sorts

The total number of times that a set of data was sorted in order to process the stmt\_operation.

Table 1093. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Statement
Application	stmt	Statement
Dynamic SQL	dynsql	Statement

**Usage** You can use this element to help identify the need for an index, since indexes can reduce the need for sorting of data. Using the related elements in the above table you can identify the SQL statement for which this element is providing sort information, and then analyze this statement to determine index candidates by looking at columns that are being sorted (for example, columns used in ORDER BY and GROUP BY clauses and join columns). See **explain** in the *Administration Guide* for information on checking whether your indexes are used to optimize sort performance.

This count includes sorts of temporary tables that were generated internally by the database manager to execute the statement. The number of sorts is associated with the first FETCH operation of the SQL statement. This information is returned to you when the operation for the statement is the first FETCH. You should note that for blocked cursors several fetches may be performed when the cursor is opened. In these cases it can be difficult to use the snapshot monitor to obtain the number of sorts, since a snapshot would need to be taken while DB2 was internally issuing the first FETCH.

A more reliable way to determine the number of sorts performed when using a blocked cursor would be with an event monitor declared for statements. The total\_sorts counter, in the statement event for the CLOSE cursor, contains the total number of sorts that were performed while executing the statement for which the cursor was defined.

---

## stmt\_source\_id - Statement source identifier

This element shows the internal identifier (ID) given to the source of the SQL statement that was run.

Table 1094. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values <sup>1</sup>	event_stmt_history	-
Deadlocks with Details History <sup>1</sup>	event_stmt_history	-
Activities	event_activitystmt	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

## Usage

You can use this element, along with **appl\_id** monitor element, to uniquely identify the origin of a request to run a particular SQL statement. You can also use this element in conjunction with other statement history entries to understand the cause of the deadlock.

---

## stmt\_start - Statement Operation Start Timestamp

The date and time when the stmt\_operation started executing.

### Element identifier

stmt\_start

### Element type

timestamp

Table 1095. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement, Timestamp
DCS Statement	dcs_stmt	Statement, Timestamp

**Usage** You can use this element with stmt\_stop to calculate the elapsed statement operation execution time.

---

## stmt\_stop - Statement Operation Stop Timestamp

The date and time when the stmt\_operation stopped executing.

### Element identifier

stmt\_stop

### Element type

Timestamp

Table 1096. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement, Timestamp
DCS Statement	dcs_stmt	Statement, Timestamp

**Usage** You can use this element with `stmt_start` to calculate the elapsed statement operation execution time.

---

## stmt\_sys\_cpu\_time - System CPU Time used by Statement

The total *system* CPU time (in seconds and microseconds) used by the currently executing statement.

**Element identifier**

`stmt_sys_cpu_time`

**Element type**

time

*Table 1097. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Statement, Timestamp
Application	stmt	Statement, Timestamp

**Usage** This element along with the other related CPU-time elements can help you understand the level of activity within an application, and can help you identify applications that could benefit from additional tuning.

This counter includes time spent on both SQL and non-SQL statements, as well as any unfenced user defined functions (UDF) or stored procedures executed by the application.

System CPU represents the time spent in system calls. User CPU represents time spent executing database manager code.

**Note:** If this information is not available for your operating system, this element will be set to 0.

---

## stmt\_text - SQL statement text monitor element

The text of the SQL statement.

*Table 1098. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	Always collected

*Table 1099. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement
Dynamic SQL	dynsql	Basic
DCS Statement	dcs_stmt	Statement

Table 1100. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details <sup>1</sup>	event_detailed_dlconn	-
Deadlocks with Details History <sup>1</sup>	event_stmt_history	-
Statements	event_stmt	-
Activities	event_activitystmt	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

## Usage

For application snapshots, this statement text helps you identify what the application was executing when the snapshot was taken, or most recently processed if no statement was being processed right at the time the snapshot was taken.

The information returned by this element is taken from the SQL statement cache and it might not be available if the cache has overflowed. The only guaranteed way to capture the SQL text of a statement is to use an event monitor for statements.

For dynamic SQL statements, this element identifies the SQL text associated with a package.

For statement event monitors, this element is returned only for dynamic statements. If a statement event monitor record cannot fit into the size of the buffer specified by the BUFFERSIZE option of a statement event monitor, the value of the **stmt\_text** monitor may be truncated so that the record can fit.

For the EVENT\_STMT\_HISTORY event monitor, this element is returned only for dynamic statements. For remaining event monitors, **stmt\_text** is returned for dynamic and static statements only if it is available in the SQL statement cache.

For information on how to query the system catalog tables to obtain static SQL statement text that is not provided due to performance considerations, see the **section\_number** monitor element.

---

## stmt\_type - Statement type monitor element

The type of statement processed.

Table 1101. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	stmt	Statement

Table 1102. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details <sup>1</sup>	event_detailed_dlconn	-
Statements	event_stmt	-
Activities	event_activitystmt	-

**1** This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

## Usage

You can use this element to determine the type of statement that is executing. It can be one of the following:

- A static SQL statement
- A dynamic SQL statement
- An operation other than an SQL statement; for example, a bind or pre-compile operation.

For the snapshot monitor, this element describes the statement that is currently being processed or was most recently processed.

**Note:** API users should refer to the sqlmon.h header file containing definitions of database system monitor constants.

---

## stmt\_usr\_cpu\_time - User CPU Time used by Statement

The total *user* CPU time (in seconds and microseconds) used by the currently executing statement.

### Element identifier

stmt\_usr\_cpu\_time

### Element type

time

Table 1103. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Statement, Timestamp
Application	stmt	Statement, Timestamp

**Usage** This element along with the other related CPU-time elements can help you understand the level of activity within an application, and can help you identify applications that could benefit from additional tuning.

This counter includes time spent on both SQL and non-SQL statements, as well as any unfenced user-defined functions (UDFs) or stored procedures executed by the application.

System CPU represents the time spent in system calls. User CPU represents time spent executing database manager code.

**Note:** If this information is not available for your operating system, this element will be set to 0.

---

## stmt\_value\_data - Value data

This element contains a string representation of a data value to an SQL statement. LOB, LONG, and structured type parameters appear as empty strings. Date, time, and timestamp fields are recorded in ISO format.

*Table 1104. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values <sup>1</sup>	stmt_value_data	-
Activities	event_activityvals	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

### Usage

You can use this element in conjunction with other statement history entries to understand the cause of the deadlock.

---

## stmt\_value\_index - Value index

This element represents the position of the input parameter marker or host variable used in the SQL statement.

*Table 1105. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values <sup>1</sup>	stmt_value_data	-
Activities	event_activityvals	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

### Usage

You can use this element in conjunction with other statement history entries to understand the cause of the deadlock.

---

## stmt\_value\_isnull - Value has null value monitor element

This element shows whether a data value associated with an SQL statement is the NULL value.

Table 1106. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values <sup>1</sup>	stmt_value_isnull	-
Activities	event_activityvals	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

### Usage

You can use this element in conjunction with other statement history entries to understand the cause of the deadlock.

---

## stmt\_value\_isreopt - Variable used for statement reoptimization monitor element

This element shows whether the provided value was a value used during statement reoptimization. It returns a value of “True” if the statement was reoptimized (for example, due to the setting of the REOPT bind option) and if the value was used as input to the SQL compiler during this reoptimization.

Table 1107. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks with Details History Values <sup>1</sup>	event_data_value	-
Activities	event_activityvals	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

### Usage

You can use this element in conjunction with the provided compilation environment to allow for full analysis of the SQL compiler’s treatment of the SQL statement.

---

## stmt\_value\_type - Value type monitor element

This element contains a string representation of the type of a data value associated with an SQL statement.

Table 1108. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-



Table 1108. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks with Details History Values <sup>1</sup>	stmt_value_type	-
Activities	event_activityvals	-

- 1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

## Usage

You can use this element in conjunction with other statement history entries to understand the cause of the deadlock.

---

## sto\_path\_free\_sz - Automatic Storage Path Free Space

This element shows the amount of free space available on a file system pointed to by a storage path. If multiple storage paths point to the same file system, the free size is not divided among them.

### Element identifier

fs\_free\_size

### Element type

information

Table 1109. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	db_sto_path_info	Buffer Pool

**Usage** You can use this element together with the following elements to gather per-node data on space utilization for the database:

- db\_storage\_path
- fs\_used\_size
- fs\_total\_size
- fs\_id
- fs\_type

---

## stop\_time - Event Stop Time

The date and time when the statement stopped executing.

Table 1110. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_stmt	Timestamp

**Usage** You can use this element with *start\_time* to calculate the elapsed statement execution time.

For a FETCH statement event, this is the time of the last successful fetch.

Note: When the Timestamp switch is OFF, this element reports "0".

---

## stored\_proc\_time - Stored Procedure Time

This element contains the aggregate amount of time, in milliseconds, that it has taken this data source to respond to stored procedure statements from all applications or a single application running on this federated server instance from the start of the federated server instance or the last reset of the database monitor counters.

Table 1111. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Timestamp
Application	appl_remote	Timestamp

For snapshot monitoring, this counter can be reset.

The response time is measured as the difference between the time the federated server submits a stored procedure to the data source, and the time it takes the data source to respond, indicating that the stored procedure has been processed.

**Usage** Use this element to determine how much actual time is spent at this data source processing stored procedures.

---

## stored\_procs - Stored Procedures

This element contains a count of the total number of stored procedures from the start of the federated server instance, or the last reset of the database monitor counters, that the federated server has called at this data source on behalf of any application.

Table 1112. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to determine how many stored procedure calls were made locally at the federated database or by an application against the federated database.

---

## sync\_runstats – Total number of synchronous RUNSTATS activities monitor element

The total number of synchronous RUNSTATS activities triggered by real-time statistics gathering for all the applications in the database. This value includes both successful and unsuccessful synchronous RUNSTATS commands. Values reported by all the database partitions are aggregated together.

Table 1113. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Statement

For snapshot monitoring, this counter can be reset.

Table 1114. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

## Usage

Use this monitor element to determine how many synchronous RUNSTATS activities have been triggered by real-time statistics gathering in the database. This value changes frequently. In order to get a better view of the system usage, take a snapshot at specific intervals over an extended period of time. When used in conjunction with **sync\_runstats\_time**, this element can help you evaluate the performance impact of synchronous RUNSTATS activities triggered by real-time statistics gathering.

---

## sync\_runstats\_time – Total time spent on synchronous RUNSTATS activities monitor element

The total time spent on synchronous RUNSTATS activities triggered by real-time statistics gathering, in milliseconds. The synchronous RUNSTATS activities occur during query compilation. At the database level, this monitor element represents the total time spent on synchronous RUNSTATS activities for all the applications running on the database, triggered by real-time statistics gathering. At the statement level, it represents the time spent on the latest synchronous RUNSTATS activities for a particular statement, triggered by real-time statistics gathering. Values reported by all the database partitions are aggregated together.

Table 1115. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Statement
Dynamic SQL	dynsql	Statement

For snapshot monitoring, this element can be reset.

Table 1116. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Statement	event_stmt	-

## Usage

Use this element along with **sync\_runstats** to evaluate the performance impact of synchronous RUNSTATS activities triggered by real-time statistics gathering, at the database level,

For dynamic SQL snapshot monitor, use this element along with **total\_exec\_time** and **num\_executions** to evaluate the impact of synchronous RUNSTATS on query performance.

For the statement event monitor, use this element along with **stmt\_start** and **stmt\_stop** for further evaluation of the impact of real-time statistics gathering.

---

## system\_cpu\_time - System CPU Time

The total *system* CPU time (in seconds and microseconds) used by the database manager agent process, the unit of work, or the statement.

When either the statement monitor switch or the timestamp switch is not turned on, this element is not collected. In that case, the monitor element displays -1 instead.

*Table 1117. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-
Transactions	event_xact	-
Statements	event_stmt	-
Activities	event_activity	-

**Usage** This element, along with the other related CPU-time elements, can help you understand the level of activity within an application, and can help you identify applications that could benefit from additional tuning.

**Note:** If this information is not available for your operating system, this element will be set to 0.

---

## tab\_file\_id - Table file ID monitor element

A file ID (FID) for the table.

*Table 1118. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLE_METRICS table function - Get table metrics	Always collected

### Usage

---

## tab\_type - Table type monitor element

This interface returns a text identifier based on defines in `sqlmon.h`, and is one of `USER_TABLE`, `DROPPED_TABLE`, `TEMP_TABLE`, `CATALOG_TABLE`, or `REORG_TABLE`.

*Table 1119. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLE_METRICS table function - Get table metrics	Always collected

### Usage

---

## table\_file\_id - Table file ID monitor element

This is the file ID (FID) for the table.

*Table 1120. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLE table function - Get table metrics	Always collected

*Table 1121. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Lock
Table	table	Basic
Lock	appl_lock_list	Lock
Lock	lock	Lock

*Table 1122. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Deadlocks	lock	-

**Usage** This element is provided for information purposes only. It is returned for compatibility with previous versions of the database system monitor, and it may *not* uniquely identify the table. Use **table\_name** and **table\_schema** monitor elements to identify the table.

---

## table\_name - Table name monitor element

The name of the table.

*Table 1123. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_TABLE table function - Get table metrics	Always collected
MON_GET_INDEX table function - Get index metrics	Always collected

*Table 1124. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table	Basic
Application	appl	Lock
Lock	appl_lock_list	Lock
Lock	lock	Lock
Lock	lock_wait	Lock

*Table 1125. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-

Table 1125. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Tables	event_table	-
Deadlocks <sup>1</sup>	lock	-
Deadlocks <sup>1</sup>	event_dlconn	-
Deadlocks with Details <sup>1</sup>	event_detailed_dlconn	-

1 This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

## Usage

Along with **table\_schema**, this element can help you determine the source of contention for resources.

At the application-level, application-lock level, and deadlock-monitoring-level, this is the table that the application is waiting to lock, because it is currently locked by another application. For snapshot monitoring, this item is only valid when the “lock” monitor group information is set to ON, and when **lock\_object\_type** indicates that the application is waiting to obtain a table lock.

For snapshot monitoring at the object-lock level, this item is returned for table-level and row-level locks. The table reported at this level is the table against which this application holds these locks.

For snapshot and event monitoring at the table level, this is the table for which information has been collected. For temporary tables, the format for **table\_name** is “TEMP (*n*, *m*)”, where:

- *n* is the table space ID
- *m* is the **table\_file\_id** element

---

## table\_scans - Table scans monitor element

The number of scans on this table.

Table 1126. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLE table function - Get table metrics	Always collected

## Usage

---

## table\_schema - Table schema name monitor element

The schema of the table.

Table 1127. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLE table function - Get table metrics	Always collected

Table 1127. Table Function Monitoring Information (continued)

Table Function	Monitor Element	Collection Level
MON_GET_INDEX table function - Get index metrics	Always collected	

Table 1128. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table	Basic
Application	appl	Lock
Lock	appl_lock_list	Lock
Lock	lock	Lock
Lock	lock_wait	Lock

Table 1129. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Tables	event_table	-
Deadlocks <sup>1</sup>	lock	-
Deadlocks <sup>1</sup>	event_dlconn	-
Deadlocks with Details <sup>1</sup>	event_detailed_dlconn	-

- <sup>1</sup> This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

## Usage

Along with **table\_name**, this element can help you determine the source of contention for resources.

For application-level, application-lock-level, deadlock-monitoring-level, this is the schema of the table that the application is waiting to lock, because it is currently locked by another application. This element is only set if **lock\_object\_type** indicates that the application is waiting to obtain a table lock. For snapshot monitoring at the application-level and application-lock levels, this item is only valid when the “lock” monitor group information is set to ON.

For snapshot monitoring at the object-lock level, this item is returned for table and row level locks. The table reported at this level is the table against which this application holds these locks.

For snapshot and event monitoring at the table level, this element identifies the schema of the table for which information has been collected. For temporary tables, the format for **table\_schema** is “<agent\_id><auth\_id>”, where:

- *agent\_id* is the Application Handle of the application creating the temporary table
- *auth\_id* is the authorization ID used by the application to connect to the database

---

## table\_type - Table type monitor element

The type of table for which information is returned.

*Table 1130. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_TABLE table function - Get table metrics	Always collected

*Table 1131. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table	Basic

*Table 1132. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Tables	event_table	-

### Usage

Use this element to help identify the table for which information is returned. If the table is a user table or a system catalog table, you can use **table\_name** and **table\_schema** to identify the table.

The type of table may be one of the following. The possible values are text strings based on defines in the sqlmon.h file.

#### USER\_TABLE

User table.

#### TEMP\_TABLE

Temporary table. Information regarding temporary tables is returned, even though the tables are not kept in the database after being used. You may still find information about this type of table useful.

#### CATALOG\_TABLE

System catalog table.

---

## tablespace\_auto\_resize\_enabled - Table space automatic resizing enabled monitor element

This element describes whether automatic resizing is enabled for the table space. A value of 1 means "Yes"; a value of 0 means "No".

*Table 1133. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

*Table 1134. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Basic



## Usage

This element is only applicable to DMS table spaces and non-temporary automatic storage table spaces. If this element is set to 1, then automatic resizing is enabled. See the following monitor elements for information about the rate of increase and the maximum size for the table space.

- `tablespace_max_size`
- `tablespace_increase_size`
- `tablespace_increase_size_percent`

---

## **tablespace\_content\_type - Table space content type monitor element**

The type of content in a table space.

*Table 1135. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

*Table 1136. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Basic

## Usage

The type of content in the table space (defined in `sqlmon.h`) can be one of the following:

- All types of permanent data.
  - Regular table space: `SQLM_TABLESPACE_CONTENT_ANY`
  - Large table space: `SQLM_TABLESPACE_CONTENT_LARGE`
- System temporary data: `SQLM_TABLESPACE_CONTENT_SYSTEMP`
- User temporary data: `SQLM_TABLESPACE_CONTENT_USRTEMP`

---

## **tablespace\_cur\_pool\_id - Buffer pool currently being used monitor element**

The buffer pool identifier for a buffer pool that a table space is currently using.

*Table 1137. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

*Table 1138. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Basic

**Usage** Each buffer pool is identified by a unique integer. The value of this element matches a value from column BUFFERPOOLID of view SYSCAT.BUFFERPOOLS.

---

## tablespace\_current\_size - Current table space size

This element shows the current size of the table space in bytes.

*Table 1139. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

**Usage** For DMS and automatic storage table spaces, this element represents the total size of all table space containers in bytes. This value is equal to the total pages for the table space (tablespace\_total\_pages) multiplied by the table space's page size (tablespace\_page\_size). This element is not applicable for SMS table spaces, or for temporary automatic storage table spaces.

On table space creation for an automatic storage table space, the current size might not match the initial size. The value of current size will be within page size multiplied by extent size multiplied by the number of storage paths of the initial size on creation (usually greater, but sometimes smaller). It will always be less than or equal to tablespace\_max\_size (if set). This is because containers can only grow by full extents, and must be grown as a set.

---

## tablespace\_extent\_size - Table space extent size monitor element

The extent size used by a table space.

*Table 1140. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

*Table 1141. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Basic

---

## tablespace\_free\_pages - Free pages in table space monitor element

The total number of pages that are currently free in a table space.

*Table 1142. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1143. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

## Usage

This is applicable only to a DMS table space.

---

## tablespace\_id - Table space identification monitor element

An integer that uniquely represents a table space used by the current database.

Table 1144. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_TABLE table function - Get table metrics	Always collected
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_EXTENT_MOVEMENT_STATUS - Get extent movement progress status metrics	Always collected

Table 1145. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Basic
Table	table	Basic

Table 1146. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Tables	event_table	-

## Usage

The value of this element matches a value from column TBSPACEID of view SYSCAT.TABLESPACES.

---

## tablespace\_increase\_size - Increase size in bytes

This element shows the size that an auto-resize table space will increase by in bytes when the table space becomes full and more space is required.

Table 1147. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

**Usage** This represents the amount of space that will be added to a table space

that can be automatically resized when it becomes full, more space is being requested, and the maximum table space size has not been reached. If the value of this element is -1 (or "AUTOMATIC" in the snapshot output), then DB2 automatically determines the value when space needs to be added. This element is only applicable to table spaces that are enabled to be automatically resized.

---

## tablespace\_increase\_size\_percent - Increase size by percent monitor element

This element shows the amount by which an auto-resize table space will increase when the table space becomes full and more space is required. The actual number of bytes is determined at the time the table space is resized based on the size of the table space at that time.

*Table 1148. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

**Usage** This represents the amount of space that will be added to a table space that can be automatically resized when it becomes full, more space is being requested, and the maximum table space size has not been reached. The growth rate is based on a percentage of the current table space size (tablespace\_current\_size) at the time the table space is resized. This element is only applicable to table spaces that are enabled to be automatically resized.

---

## tablespace\_initial\_size - Initial table space size

The initial size of the automatic storage table space in bytes.

*Table 1149. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

**Usage** For non-temporary automatic storage table spaces, this monitor element represents the initial size in bytes for the table space when it was created.

---

## tablespace\_last\_resize\_failed - Last resize attempt failed

This element describes whether or not the last attempt to automatically increase the size of the table space failed. A value of 1 means yes, 0 means no.

*Table 1150. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

**Usage** For an automatic storage table space, this element may show that there is no space left on any of the database's storage paths. For a non-automatic storage table space, a failure means that one of the containers could not be extended because its filesystem was full. Another reason for failure is that the maximum size of the table space has been reached. This element is only applicable to table spaces that are enabled to be automatically resized.

---

## tablespace\_last\_resize\_time - Time of last successful resize

This element shows a timestamp representing the last time that the size of the table space was successfully increased.

Table 1151. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

**Usage** For table spaces that can be automatically resized, this element represents the last time that space was automatically added to the table space when it became full, more space was being requested, and the maximum table space size had not been reached. This element is only applicable to table spaces that are enabled to be automatically resized.

---

## tablespace\_max\_size - Maximum table space size

This element shows the maximum size in bytes to which the table space can automatically resize or increase.

Table 1152. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

**Usage** This represents the maximum size in bytes to which a table space that can be automatically resized can automatically increase. If this value is equal to the `tablespace_current_size` element, then there is no room for the table space to grow. If the value of this element is -1, then the maximum size is considered to be “unlimited” and the table space can automatically resize until the file systems are full or the architectural size limit of the table space is reached. (This limit is described in the SQL Limits appendix of the *SQL Reference*). This element is only applicable to table spaces that are enabled for automatic resizing.

---

## tablespace\_min\_recovery\_time - Minimum Recovery Time For Rollforward

A timestamp showing the earliest point in time to which a table space can be rolled forward.

**Element identifier**

tablespace\_min\_recovery\_time

**Element type**

information

Table 1153. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

**Usage** Displayed only if non zero.

## tablespace\_name - Table space name monitor element

The name of a table space.

*Table 1154. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_EXTENT_MOVEMENT_STATUS - Get extent movement progress status metrics	Always collected

*Table 1155. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Basic
Lock	appl_lock_list	Basic
Lock	lock	Lock
Lock	lock_wait	Lock

*Table 1156. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Deadlocks <sup>1</sup>	lock	-
Deadlocks <sup>1</sup>	event_dlconn	-
Deadlocks with Details <sup>1</sup>	event_detailed_dlconn	-
Table Space	tablespace_list	-

- 1** This option has been deprecated. Its use is no longer recommended and might be removed in a future release. Use the CREATE EVENT MONITOR FOR LOCKING statement to monitor lock-related events, such as lock timeouts, lock waits, and deadlocks.

### Usage

This element can help you determine the source of contention for resources.

It is equivalent to the TBSPACE column in the database catalog table SYSCAT.TABLESPACES. At the application level, application-lock level, and deadlock monitoring level, this is the name of the table space that the application is waiting to lock. Another application currently holds a lock on this table space.

At the lock level, this is the name of the table space against which the application currently holds a lock.

At the table space level (when the buffer pool monitor group is ON), this is the name of the table space for which information is returned.

This element will not be returned for a table lock held on a partitioned table.

---

## tablespace\_next\_pool\_id - Buffer pool that will be used at next startup monitor element

The buffer pool identifier for a buffer pool that a table space will use at the next database startup.

Table 1157. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1158. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Basic

**Usage** Each buffer pool is identified by a unique integer. The value of this element matches a value from column BUFFERPOOLID of view SYSCAT.BUFFERPOOLS

---

## tablespace\_num\_containers - Number of Containers in Table Space

Total number of containers in the table space.

Table 1159. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

---

## tablespace\_num\_quiescers - Number of Quiescers

The number of users quiescing the table space (can be in the range of 0 to 5).

Table 1160. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

**Usage** This value represents the number of agents that have quiesced the table space (either in "SHARE", "UPDATE", or "EXCLUSIVE" mode). For each quiescer, the following information is returned in a tablespace\_quiescer logical data group:

- User authorization ID of the quiescer
- Agent ID of the quiescer
- Table space ID of the object that was quiesced that resulted in this table space being quiesced
- Object ID of the object that was quiesced that resulted in this table space being quiesced
- Quiesce state

---

## tablespace\_num\_ranges - Number of Ranges in the Table Space Map

The number of ranges (entries) in the table space map. This can be in the range of 1 to 100's (but is usually less than a dozen). The table space map only exists for DMS table spaces.

*Table 1161. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

---

## tablespace\_page\_size - Table space page size monitor element

Page size used by a table space in bytes.

*Table 1162. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

*Table 1163. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Basic

---

## tablespace\_page\_top - Table space high watermark monitor element

The page in a table space that is holding the high watermark.

*Table 1164. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

*Table 1165. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

### Usage

For DMS, this element represents the page number of the first free extent following the last allocated extent of a table space. Note that this is not really a "high watermark", but rather a "current watermark", since the value can decrease. For SMS, this is not applicable.

---

## tablespace\_paths\_dropped - Table space using dropped path monitor element

Indicates that the table space is using a storage path that has been dropped.



Table 1166. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1167. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

## Usage

For table spaces using automatic storage, use this monitor element to determine whether any of the table space containers reside on a storage path that has been dropped. Before storage paths are physically dropped from the database, all table spaces must stop using them. To stop using a dropped storage path, either drop the table space or rebalance the table space using the REBALANCE clause of the ALTER TABLESPACE statement.

---

## tablespace\_pending\_free\_pages - Pending free pages in table space monitor element

The number of pages in a table space which would become free if all pending transactions are committed or rolled back and new space is requested for an object.

Table 1168. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1169. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

## Usage

This is applicable only to a DMS table space.

---

## tablespace\_prefetch\_size - Table space prefetch size monitor element

The maximum number of pages the prefetcher gets from the disk at a time.

Table 1170. Table Function Monitoring Information

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

Table 1171. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Basic
Table Space	tablespace_nodeinfo	Basic

### Usage

- If automatic prefetch size is enabled, this element reports the value "-1" in the *tablespace* Logical Data Grouping, and the actual value is reported in the *tablespace\_nodeinfo* Logical Data Grouping.
- If automatic prefetch size is not enabled, this element reports the actual value in the *tablespace* Logical Data Grouping, and the element does not appear in the *tablespace\_nodeinfo* Logical Data Grouping.

---

## tablespace\_rebalancer\_extents\_processed - Number of Extents the Rebalancer has Processed

The number of extents that the rebalancer has already moved since the rebalancer has been started or restarted (whichever is most recent).

### Element identifier

tablespace\_rebalancer\_extents\_processed

### Element type

information

Table 1172. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

**Usage** This can be used as an indicator of the completion level of the rebalancer. You can monitor the progress of rebalancing by keeping track of the change in this element over time. You can use *tablespace\_state* and *rebalance\_mode* to check if the rebalancing is completed. This is only applicable to a DMS table space.

---

## tablespace\_rebalancer\_extents\_remaining - Total Number of Extents to be Processed by the Rebalancer

The number of extents to be moved. This value is calculated at either the rebalancer start time or restart time (whichever is most recent).

### Element identifier

tablespace\_rebalancer\_extents\_remaining

### Element type

information

Table 1173. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

**Usage** This can be used as an indicator of the completion level of the rebalancer. You can monitor the progress of rebalancing by keeping track of the

change in this element over time. You can use `tablespace_state` to check if rebalancing has completed. This is only applicable to a DMS table space.

---

## tablespace\_rebalancer\_last\_extent\_moved - Last Extent Moved by the Rebalancer

The last extent moved by the rebalancer.

**Element identifier**

`tablespace_rebalancer_last_extent_moved`

**Element type**

information

*Table 1174. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	<code>tablespace_nodeinfo</code>	Basic

**Usage** This can be used as an indicator of the completion level of the rebalancer. You can monitor the progress of rebalancing by keeping track of the change in this element over time. You can use `tablespace_state` and `rebalance_mode` to check if the rebalancing is completed. This is only applicable to a DMS table space.

---

## tablespace\_rebalancer\_mode - Rebalancer mode monitor element

Indicates whether the current rebalance process is removing space from a table space or adding space to a table space.

*Table 1175. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

*Table 1176. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	<code>tablespace_nodeinfo</code>	Basic

**Usage**

A *forward rebalance* occurs when new containers are added or existing containers have their size increased. In a forward rebalancing operation, data movement starts with the first extent in the table space and ends with the high watermark extent.

A *reverse rebalance* occurs when containers are removed or reduced in size and data needs to move out of the space being freed. In a reverse rebalancing operation, data movement starts at the high watermark extent and moves in reverse order through the table space, ending with the first extent in the table space.

A *two-pass rebalance* is a forward rebalance followed by a reverse rebalance. A two-pass rebalance might occur when containers are being both added and dropped as part of the rebalance operation.

For DMS non-automatic storage table spaces, this monitor element indicates the type of rebalance that is occurring for the table space. Only a single forward rebalance or a single reverse rebalance can occur for DMS non-automatic table space.

For automatic storage table spaces, this monitor element indicates what the current rebalance process is doing to the table space. In general, only a single forward rebalance or a single reverse rebalance is necessary when a rebalance is initiated. However, there are cases when a two-pass rebalance is necessary for automatic storage table spaces.

The possible `tablespace_rebalancer_mode` values are defined in the `sqlmon.h` file as follows:

**SQLM\_TABLESPACE\_NO\_REBAL**

No rebalancing is taking place.

**SQLM\_TABLESPACE\_FWD\_REBAL**

Forward rebalance is taking place.

**SQLM\_TABLESPACE\_REV\_REBAL**

Reverse rebalance is taking place.

**SQLM\_TABLESPACE\_FWD\_REBAL\_OF\_2PASS**

The forward rebalance phase of two pass rebalance is taking place.

**SQLM\_TABLESPACE\_REV\_REBAL\_OF\_2PASS**

The reverse rebalance phase of two pass rebalance is taking place.

## **tablespace\_rebalancer\_priority - Current Rebalancer Priority**

The priority at which the rebalancer is running in the database.

*Table 1177. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

**Usage** This is only applicable to a DMS table space.

## **tablespace\_rebalancer\_restart\_time - Rebalancer Restart Time**

A timestamp representing when a rebalancer was restarted after being paused or stopped.

**Element identifier**

tablespace\_rebalancer\_restart\_time

**Element type**

information

*Table 1178. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

**Usage** This can be used as an indicator of the completion level of the rebalancer. It will note when the rebalancer was restarted, and will allow for the derivation of the speed of the rebalancer and the estimated time until completion. This is only applicable to a DMS table space.

---

## tablespace\_rebalancer\_start\_time - Rebalancer Start Time

A timestamp representing when a rebalancer was initially started.

**Element identifier**

tablespace\_rebalancer\_start\_time

**Element type**

information

*Table 1179. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

**Usage** This will be used to note the time at which a rebalancer was initially started. This can be used to derive metrics as to the speed at which the rebalancer is operating, and the estimated time of completion of the rebalance. This is only applicable to a DMS table space.

---

## tablespace\_state - Table space state monitor element

This element describes the current state of a table space.

*Table 1180. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

*Table 1181. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

### Usage

This element contains a hexadecimal value indicating the current table space state. The externally visible state of a table space is composed of the hexadecimal sum of certain state values. For example, if the state is "quiesced: EXCLUSIVE" and "Load pending", the value is 0x0004 + 0x0008, which is 0x000c. Use the db2tbst command to obtain the table space state associated with a given hexadecimal value.

*Table 1182. Bit definitions listed in sqlutil.h*

Hexadecimal Value	Decimal Value	State
0x0	0	Normal (see the definition SQLB_NORMAL in sqlutil.h)
0x1	1	Quiesced: SHARE
0x2	2	Quiesced: UPDATE
0x4	4	Quiesced: EXCLUSIVE
0x8	8	Load pending
0x10	16	Delete pending
0x20	32	Backup pending
0x40	64	Roll forward in progress

Table 1182. Bit definitions listed in sqlutil.h (continued)

Hexadecimal Value	Decimal Value	State
0x80	128	Roll forward pending
0x100	256	Restore pending
0x100	256	Recovery pending (not used)
0x200	512	Disable pending
0x400	1024	Reorg in progress
0x800	2048	Backup in progress
0x1000	4096	Storage must be defined
0x2000	8192	Restore in progress
0x4000	16384	Offline and not accessible
0x8000	32768	Drop pending
0x80000	524288	Move in progress
0x2000000	33554432	Storage may be defined
0x4000000	67108864	Storage Definition is in 'final' state
0x8000000	134217728	Storage Definition was changed prior to rollforward
0x10000000	268435456	DMS rebalancer is active
0x20000000	536870912	TBS deletion in progress
0x40000000	1073741824	TBS creation in progress

## tablespace\_state\_change\_object\_id - State Change Object Identification

The object that caused the table space state to be set to "Load pending" or "Delete pending".

### Element identifier

tablespace\_state\_change\_object\_id

### Element type

information

Table 1183. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

**Usage** This element is meaningful only if the table space state is "Load pending" or "Delete pending". If nonzero, the value of this element matches a value from column TABLEID of view SYSCAT.TABLES.

## tablespace\_state\_change\_ts\_id - State Change Table Space Identification

If the table space state is "Load pending" or "Delete pending", this shows the table space ID of the object that caused the table space state to be set.

### Element identifier

tablespace\_state\_change\_ts\_id

**Element type**  
information

*Table 1184. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic

**Usage** This element is meaningful only if the table space state is "Load pending" or "Delete pending". If nonzero, the value of this element matches a value from column TABLESPACEID of view SYSCAT.TABLES.

---

## tablespace\_total\_pages - Total pages in table space monitor element

Total number of pages in a table space.

*Table 1185. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

*Table 1186. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic (DMS table spaces)  Buffer Pool (SMS table spaces)

### Usage

Total operating system space occupied by a table space. For DMS, this is the sum of the container sizes (including overhead). For SMS, this is the sum of all file space used for the tables stored in this table space (and is only collected if the buffer pool switch is on).

---

## tablespace\_type - Table space type monitor element

The type of a table space.

*Table 1187. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

*Table 1188. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Basic

## Usage

This element shows whether this table space is a database managed table space (DMS), or system managed table space (SMS).

The values for `tablespace_type` (defined in `sqlmon.h`) are as follows:

- For DMS: `SQLM_TABLESPACE_TYP_DMS`
- For SMS: `SQLM_TABLESPACE_TYP_SMS`

---

## tablespace\_usable\_pages - Usable pages in table space monitor element

The total number of pages in a table space minus overhead pages.

*Table 1189. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

*Table 1190. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic (DMS table spaces) Buffer Pool (SMS table spaces)

## Usage

This element is applicable to DMS table spaces only. For SMS table spaces, this element will have the same value as the **tablespace\_total\_pages** monitor element.

During a table space rebalance, the number of usable pages will include pages for the newly added container, but these new pages may not be reflected in the number of free pages until the rebalance is complete. When a table space rebalance is not taking place, the number of used pages plus the number of free pages, plus the number of pending free pages will equal the number of usable pages.

---

## tablespace\_used\_pages - Used pages in table space monitor element

The total number of pages that are currently used (not free) in a table space.

*Table 1191. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

*Table 1192. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace_nodeinfo	Basic (DMS table spaces) Buffer Pool (SMS table spaces)



## Usage

This is the total number of pages in use for a DMS table space. For an SMS table space it is equal to the value of `tablespace_total_pages` monitor element.

---

### **tablespace\_using\_auto\_storage - Table space enabled for automatic storage monitor element**

This element describes whether the table space was created as an automatic storage table space. A value of 1 means "Yes"; a value of 0 means "No".

*Table 1193. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

*Table 1194. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	tablespace	Basic

## Usage

You can use this element to determine whether the given table space was created using automatic storage (that is, created with the `MANAGED BY AUTOMATIC STORAGE` clause), rather than with containers that are explicitly provided. The table space can have containers that exist on some or all of the storage paths associated with the database.

---

### **tbsp\_max\_page\_top - Maximum table space page high watermark monitor element**

The highest allocated page number for a DMS table space since the database was activated.

*Table 1195. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

## Usage

This value changes whenever the value of the `tablespace_page_top` monitor element increases.

---

### **tcPIP\_recv\_volume - TCP/IP received volume monitor element**

The amount of data received by the data server from clients over TCP/IP. This value is reported in bytes.

Table 1196. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1197. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## tcpip\_rcv\_wait\_time - TCP/IP received wait time monitor element

The time spent waiting for an incoming client request over TCP/IP excluding idle time. The value is given in milliseconds.

Table 1198. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 1198. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1199. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## tcpip\_recvs\_total - TCP/IP receives total monitor element

The number of times data was received by the database server from the client application over TCP/IP.

Table 1200. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1200. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1201. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## tcpip\_send\_volume - TCP/IP send volume monitor element

The amount of data sent by data server to client. This value is reported in bytes.

Table 1202. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1203. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE

Table 1203. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## tcPIP\_send\_wait\_time - TCP/IP send wait time monitor element

Time spent blocking on a TCP/IP send to the client. The value is given in milliseconds.

Table 1204. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1205. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## tcPIP\_sends\_total - TCP/IP sends total monitor element

The number of times data was sent from the database server to the client application over TCP/IP.

Table 1206. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1207. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## temp\_tablespace\_top - Temporary table space top monitor element

The high watermark in KB for the temporary table space usage of DML activities at all nesting levels in a service class or work class. For service classes, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the service class is set to NONE. For work classes, this monitor elements returns -1 if no COLLECT AGGREGATE ACTIVITY DATA work action is specified for the work class. For workloads, this monitor element returns -1 when COLLECT AGGREGATE ACTIVITY DATA for the workload is set to NONE.

For service classes, when you remap activities between service subclasses with a REMAP ACTIVITY action, only the temp\_tablespace\_top high watermark of the service subclass where an activity completes is changed. High watermarks of service subclasses an activity is mapped to but does not complete in are unaffected.

Table 1208. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats	-
Statistics	event_wcstats	-
Statistics	event_wlstats	-

## Usage

Use this element to determine the highest DML activity system temporary table space usage reached on a partition for a service class, workload, or work class in the time interval collected.

This element is only updated by activities that have a temporary table space threshold applied to them. If no temporary table space threshold is applied to an activity, a value of 0 is returned.

---

## territory\_code - Database Territory Code

The territory code of the database for which the monitor data is collected. This monitor element was formerly known as country\_code.

Table 1209. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
Application	appl	Basic

Table 1210. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Event Log Header	event_log_header	-
Connections	event_connheader	-

**Usage** Territory code information is recorded in the database configuration file.

For DRDA AS connections, this element will be set to 0.

---

## threshold\_action - Threshold action monitor element

The action of the threshold to which this threshold violation record applies. Possible values include Stop, Continue and Remap.

Table 1211. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Threshold violations	event_thresholdviolations	-

## Usage

Use this element to determine whether the activity that violated the threshold was stopped when the violation occurred, was allowed to continue executing, or was remapped to another service subclass. If the activity was stopped, the application that submitted the activity will have received an SQL4712N error. If the activity

was remapped to another service subclass, agents working for the activity on the partition will be moving to the target service subclass of the threshold.

---

## threshold\_domain - Threshold domain monitor element

The domain of the threshold responsible for this queue.

Possible values are

- Database
- Work Action Set
- Service Superclass
- Service Subclass
- Workload

Table 1212. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_qstats	-

### Usage

This element can be used for distinguishing the queue statistics of thresholds that have the same predicate but different domains.

---

## threshold\_maxvalue - Threshold maximum value monitor element

For non-queuing thresholds, this monitor element represents the value that was exceeded to cause this threshold violation. For queuing thresholds, this monitor element represents the level of concurrency that caused the queuing. The level of concurrency that caused the violation of the queuing threshold is the sum of **threshold\_maxvalue** and **threshold\_queuesize** monitor elements.

Table 1213. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Threshold violations	event_thresholdviolations	-

### Usage

For activity thresholds, this element provides a historical record of what the threshold's maximum value was at the time the threshold was violated. This is useful when the threshold's maximum value has changed since the time of the violation and the old value is no longer available from the SYSCAT.THRESHOLDS view.

---

## threshold\_name - Threshold name monitor element

The unique name of the threshold responsible for this queue.

Table 1214. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_qstats	-



## Usage

Use this element to uniquely identify the queuing threshold whose statistics this record represents.

---

## threshold\_predicate - Threshold predicate monitor element

Identifies the type of threshold that was violated or for which statistics were collected.

*Table 1215. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Threshold violations	event_thresholdviolations	-
Statistics	event_qstats	-

## Usage

Use this monitor element in conjunction with other statistics or threshold violation monitor elements for analysis of a threshold violation.

---

## threshold\_queuesize - Threshold queue size monitor element

The size of the queue for a queuing threshold. An attempt to exceed this size causes a threshold violation. For a non-queuing threshold, this value is 0.

*Table 1216. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Threshold violations	event_thresholdviolations	-

## Usage

Use this element to determine the number of activities or connections in the queue for this threshold at the time the threshold was violated.

---

## thresholdid - Threshold ID monitor element

Identifies the threshold to which a threshold violation record applies or for which queue statistics were collected.

*Table 1217. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Threshold violations	event_thresholdviolations	-
Statistics	event_qstats	-

## Usage

Use this monitor element in conjunction with other activity history monitor elements for analysis of a threshold queue or for analysis of the activity that violated a threshold.

---

## time\_completed - Time completed monitor element

The time at which the activity described by this activity record finished executing. This element is a local timestamp.

This field has a value of "0000-00-00-00.00.00.000000" when a full activity record could not be written to a table event monitor due to memory limitations or if the activity was captured while it was in progress.

### Element identifier

time\_completed

### Element type

information

-->

Table 1218. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

## Usage

Use this element in conjunction with other activity history elements for analysis of the behavior of an activity.

---

## time\_created - Time created monitor element

The time at which a user submitted the activity described by this activity record. This element is a local timestamp.

Table 1219. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

## Usage

Use this element in conjunction with other activity history elements for analysis of the behavior of an activity.

---

## time\_of\_violation - Time of violation monitor element

The time at which the threshold violation described in this threshold violation record occurred. This element is a local timestamp.

Table 1220. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Threshold violations	event_thresholdviolations	-

## Usage

Use this element in conjunction with other threshold violations monitor elements for analysis of a threshold violation.

---

## time\_stamp - Snapshot Time

The date and time when the database system monitor information was collected.

*Table 1221. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	collected	Basic

**Usage** You can use this element to help relate data chronologically if you are saving the results in a file or database for ongoing analysis.

---

## time\_started - Time started monitor element

The time at which the activity described by this activity record began executing. This element is a local timestamp.

*Table 1222. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-

### Usage

Use this element in conjunction with other activity history elements for analysis of the behavior of an activity.

---

## time\_zone\_disp - Time Zone Displacement

Number of seconds that the local time zone is displaced from Greenwich Mean Time (GMT).

*Table 1223. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	collected	Basic

**Usage** All time reported by reported by the database system monitor is GMT, this displacement calculates the local time.

---

## top - Histogram bin top monitor element

The inclusive top end of the range of a histogram bin. The value of this monitor element is also the bottom exclusive end of the range of the next histogram bin.

*Table 1224. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_histogrambin	-

### Usage

Use this element with the corresponding **bottom** element to determine the range of a bin within a histogram.

---

## tot\_log\_used\_top - Maximum Total Log Space Used

The maximum amount of total log space used (in bytes).

Table 1225. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

Table 1226. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

**Usage** You can use this element to help evaluate the amount of primary log space that you have allocated. Comparing the value of this element with the amount of primary log space you have allocated can help you to evaluate your configuration parameter settings. Your primary log space allocation can be calculated using the following formula:

$$\text{logprimary} \times \text{logfilsiz} \times 4096 \text{ (see note below)}$$

You can use this element in conjunction with *sec\_log\_used\_top* and *sec\_logs\_allocated* to show your current dependency on secondary logs.

This value includes space used in both primary and secondary log files.

You may need to adjust the following configuration parameters:

- logfilsiz
- logprimary
- logsecond

**Note:** While the database system monitor information is given in bytes, the configuration parameters are set in pages, which are each 4K bytes.

---

## total\_act\_time - Total activity time monitor element

The total amount of time spent executing activities. This value is given in milliseconds.

Table 1227. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

---

## total\_act\_wait\_time - Total activity wait time monitor element

Total time spent waiting within the DB2 database server, while processing an activity. The value is given in milliseconds.

Table 1228. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	COLLECT ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

## Usage

Use this element along with the **total\_act\_request\_time** element to determine the percentage of time the data server spends working on the activity.

$$\frac{(\text{total\_act\_request\_time} - \text{total\_act\_wait\_time})}{(\text{total\_act\_request\_time})} = \text{\% of time data server is actively working on activity}$$

## total\_app\_rqst\_time - Total application request time monitor element

The total elapsed time spent on application requests; this is the total time spent by coordinator agents on the server executing application requests.

Table 1229. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

## Usage

Use this monitor element to determine the time that the application request spent in the DB2 data server. This value can be used to help determine if the data server is the source of an observed performance problem.

For example, if a user reports that there is a problem with an application and it has taken 20 minutes to return, and if you determine that total application request time is 1 minute and there are currently no application requests in progress for the connection, then the performance problem might lie outside of the DB2 data server.

---

## total\_buffers\_rcvd - Total FCM Buffers Received

The total number of FCM buffers received by the node issuing the GET SNAPSHOT command from the node identified by the *node\_number* (see the *db2nodes.cfg* file).

*Table 1230. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	fcm_node	Basic

**Usage** You can use this element to measure the level of traffic between the current node and the remote node. If the total number of FCM buffers received from this node is high, you may want to redistribute the database, or move tables to reduce the inter-node traffic.

---

## total\_buffers\_sent - Total FCM Buffers Sent

The total number of FCM buffers that have been sent from the node issuing the GET SNAPSHOT command to the node identified by the *node\_number* (see the *db2nodes.cfg* file).

*Table 1231. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	fcm_node	Basic

**Usage** You can use this element to measure the level of traffic between the current node and the remote node. If the total number of FCM buffers sent to this node is high, you may want to redistribute the database, or move tables to reduce the inter-node traffic.

---

## total\_cons - Connects Since Database Activation

Indicates the number of connections to the database since the first connect, activate, or last reset (coordinator agents).

*Table 1232. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Database	dbase_remote	Basic

For snapshot monitoring, this counter can be reset.

*Table 1233. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-

**Usage** You can use this element in conjunction with the *db\_conn\_time* and the

db2start\_time monitor elements to calculate the frequency at which applications have connected to the database.

If the frequency of connects is low, you may want to explicitly activate the database using the ACTIVATE DATABASE command before connecting any other application, because of the extra overhead that is associated with the first connect to a database (for example, initial buffer pool allocation). This will result in subsequent connects being processed at a higher rate.

**Note:** When you reset this element, its value is set to the number of applications that are currently connected, not to zero.

---

## total\_cpu\_time - Total CPU time monitor element

The total amount of CPU time used while within DB2. Represents total of both user and system CPU time. This value is given in microseconds.

*Table 1234. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

*Table 1235. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE

Table 1235. Event Monitoring Information (continued)

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-

## total\_exec\_time - Elapsed Statement Execution Time

The total time in seconds and microseconds that was spent executing a particular statement in the SQL cache.

Table 1236. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Dynamic SQL	dynsql	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element with num\_executions determine the average elapsed time for the statement and identify the SQL statements that would most benefit from a tuning of their SQL. The num\_compilation must be considered when evaluating the contents of this element.

## total\_move\_time - Total extent move time monitor element

In milliseconds, the total move time for all extents moved during the table space rebalance process.

Table 1237. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_EXTENT_MOVEMENT_STATUS - Get extent movement progress status metrics	Always collected

## total\_hash\_joins - Total Hash Joins

The total number of hash joins executed.

**Element identifier**  
total\_hash\_joins

**Element type**  
counter

Table 1238. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.



Table 1239. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** At the database or application level, use this value in conjunction with `hash_join_overflows` and `hash_join_small_overflows` to determine if a significant percentage of hash joins would benefit from modest increases in the sort heap size.

---

## total\_hash\_loops - Total Hash Loops

The total number of times that a single partition of a hash join was larger than the available sort heap space.

Table 1240. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 1241. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** Values for this element indicate inefficient execution of hash joins. This might indicate that the sort heap size is too small or the sort heap threshold is too small. Use this value in conjunction with the other hash join variables to tune the sort heap size (*sortheap*) and sort heap threshold (*sheapthres*) configuration parameters.

---

## total\_log\_available - Total Log Available

The amount of active log space in the database that is not being used by uncommitted transactions (in bytes).

Table 1242. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

### Usage

Use this element in conjunction with `total_log_used` to determine whether you may need to adjust the following configuration parameters to avoid running out of log space:

- `logfilesiz`
- `logprimary`
- `logsecond`

If `total_log_available` goes down to 0, SQL0964N will be returned. You may need to increase the above configuration parameters, or end the oldest transaction by COMMIT, ROLLBACK or FORCE APPLICATION.

If `logsecond` is set to -1 this element will contain `SQLM_LOGSPACE_INFINITE`.

**Note:** While the database system monitor information is given in bytes, the configuration parameters are set in pages, which are each 4K bytes.

---

## total\_log\_used - Total Log Space Used

The total amount of active log space currently used (in bytes) in the database.

*Table 1243. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

**Usage** Use this element in conjunction with `total_log_available` to determine whether you may need to adjust the following configuration parameters to avoid running out of log space:

- `logfilsiz`
- `logprimary`
- `logsecond`

**Note:** While the database system monitor information is given in bytes, the configuration parameters are set in pages, which are each 4K bytes.

---

## total\_olap\_funcs - Total OLAP Functions monitor element

The total number of OLAP functions executed.

*Table 1244. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

*Table 1245. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

### Usage

At the database or application level, use this value in conjunction with `olap_func_overflows` to determine if a significant percentage of OLAP functions would benefit from modest increases in the sort heap size.

---

## total\_rqst\_mapped\_in - Total request mapped-in monitor element

The total number of requests that were mapped into this service subclass via a remap threshold or a work action set.

*Table 1246. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

*Table 1247. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE

---

## total\_rqst\_mapped\_out - Total request mapped-out monitor element

The total number of requests that were mapped out of this service subclass via a remap threshold or a work action set.

*Table 1248. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

*Table 1249. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE

---

## total\_rqst\_time - Total request time monitor element

The total amount of time spent working on requests. This value is reported in milliseconds.

*Table 1250. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1250. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

## Usage

### total\_sec\_cons - Secondary Connections

The number of connections made by a subagent to the database at the node.

**Element identifier**

total\_sec\_cons

**Element type**

counter

Table 1251. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic

**Usage** You can use this element in conjunction with the total\_cons, db\_conn\_time, and the db2start\_time monitor elements to calculate the frequency at which applications have connected to the database.

### total\_section\_sorts - Total section sorts monitor element

Total number of sorts performed during section execution, which is the execution of the compiled query plan generated by the SQL statement that was issued by the client application.

Table 1252. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1252. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

## Usage

Use this element with `total_section_sort_time` to calculate the average amount of time spent performing sorts during section execution.

At the activity and package cache levels, use this element to identify statements which are performing large numbers of sorts. These statements may benefit from additional tuning to reduce the number of sorts. You can also use the `EXPLAIN` statement to identify the number of sorts a statement performs.

---

## total\_section\_sort\_proc\_time - Total section sort processing time monitor element

Total amount of processing (non-wait) time spent performing sorts while executing a section, which is the execution of the compiled query plan generated by the SQL statement that was issued by the client application.

Table 1253. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 1253. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

## Usage

At the system level, use this element with `total_section_sorts` to calculate the average sort processing time (does not include waits) during section execution, which can indicate whether or not sorting is an issue as far as performance is concerned.

At the activity level, use this element to identify statements that spend a large amount of time sorting. These statements may benefit from additional tuning to reduce the sort time.

---

## **total\_section\_sort\_time - Total section sort time monitor element**

Total amount of time spent performing sorts while executing a section, which is the execution of the compiled query plan generated by the SQL statement that was issued by the client application.

Table 1254. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE

Table 1254. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

## Usage

At the system level, use this element with `total_section_sorts` to calculate the average sort time during section execution, which can indicate whether or not sorting is an issue as far as statement performance is concerned.

**Note:** The `total_section_sort_time` element includes both wait and processing time. If  $(total\_section\_sort\_time - total\_section\_sort\_proc\_time)$  is high, sorts are spending a lot of time waiting. For example, if sorts are frequently spilling to disk, the `total_section_sort_time` will increase due to I/O waits. This time will not show up in `total_section_sort_proc_time` which only counts the time actively processing a sort. In this case, you may consider tuning sort memory to improve performance.

At the activity level, use this element to identify statements that spend a large amount of time sorting. These statements may benefit from additional tuning to reduce the sort time.

---

## total\_sort\_time - Total sort time monitor element

The total elapsed time for all sorts that have been executed. This value is reported in milliseconds.

Table 1255. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Sort
Application	appl	Sort
Application	stmt	Sort
Dynamic SQL	dynsql	Sort

For snapshot monitoring, this counter can be reset.

*Table 1256. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-
Statements	event_stmt	-
Activities	event_activity	Statement, Sort

## Usage

At a database or application level, use this element with **total\_sorts** to calculate the average sort time, which can indicate whether or not sorting is an issue as far as performance is concerned.

At a statement level, use this element to identify statements that spend a lot of time sorting. These statements may benefit from additional tuning to reduce the sort time.

This count also includes sort time of temporary tables created during related operations. It provides information for one statement, one application, or all applications accessing one database.

When using monitor elements providing elapsed times, you should consider:

1. Elapsed times are affected by system load, so the more processes you have running, the higher this elapsed time value.
2. To calculate this monitor element at a database level, the database system monitor sums the application-level times. This can result in double counting elapsed times at a database level, since more than one application process can be running at the same time.

To provide meaningful data from the database level, you should normalize the data to a lower level. For example:

```
total_sort_time / total_sorts
provides information about the average elapsed time for each sort.
```

---

## total\_sorts - Total sorts monitor element

The total number of sorts that have been executed.

*Table 1257. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE



Table 1257. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

Table 1258. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 1259. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Connection	event_conn	-
Statements	event_stmt	-
Activities	event_activity	Statement, Sort

## Usage

At a database or application level, use this value with **sort\_overflows** to calculate the percentage of sorts that need more heap space. You can also use it with **total\_sort\_time** to calculate the average sort time.

If the number of sort overflows is small with respect to the total sorts, then increasing the sort heap size may have little impact on performance, unless this buffer size is increased substantially.

At a statement level, use this element to identify statements which are performing large numbers of sorts. These statements may benefit from additional tuning to reduce the number of sorts. You can also use the SQL EXPLAIN statement to identify the number of sorts a statement performs.

---

## total\_sys\_cpu\_time - Total system CPU time for a statement monitor element

The total system CPU time for an SQL statement.

*Table 1260. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Dynamic SQL	dynsql	Statement

For snapshot monitoring, this counter can be reset.

**Usage** Use this element with Elapsed Statement Execution Time and Total User CPU for a Statement to evaluate which statements are the most expensive.

---

## total\_sorts - Total sorts monitor element

The total number of sorts that have been executed.

*Table 1261. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1261. Table Function Monitoring Information (continued)

Table Function	Monitor Element	Collection Level
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS	BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS	BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS	BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS	BASE

Table 1262. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 1263. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Database	event_db	-
Connection	event_conn	-
Statements	event_stmt	-
Activities	event_activity	Statement, Sort

## Usage

At a database or application level, use this value with **sort\_overflows** to calculate the percentage of sorts that need more heap space. You can also use it with **total\_sort\_time** to calculate the average sort time.

If the number of sort overflows is small with respect to the total sorts, then increasing the sort heap size may have little impact on performance, unless this buffer size is increased substantially.

At a statement level, use this element to identify statements which are performing large numbers of sorts. These statements may benefit from additional tuning to reduce the number of sorts. You can also use the SQL EXPLAIN statement to

identify the number of sorts a statement performs.

---

## **total\_usr\_cpu\_time - Total user CPU time for a statement monitor element**

The total user CPU time for an SQL statement.

*Table 1264. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Dynamic SQL	dynsql	Statement

For snapshot monitoring, this counter can be reset.

### **Usage**

Use this element with Elapsed Statement Execution Time and to evaluate the longest running statements.

---

## **total\_wait\_time - Total wait time monitor element**

The total time spent waiting within the DB2 database server. The value is given in milliseconds.

*Table 1265. Table Function Monitoring Information*

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

### **Usage**

To understand the percentage of time the database server spends actively working on requests, use the following ratio:

$(\text{total\_rqst\_time} - \text{total\_wait\_time}) / \text{total\_rqst\_time}$

The value of the **client\_idle\_wait\_time** monitor element is not included in the value of the **total\_wait\_time** monitor element. The **total\_wait\_time** element represents only time spent waiting while the database server is processing requests.

---

## tpmon\_acc\_str - TP monitor client accounting string monitor element

The data passed to the target database for logging and diagnostic purposes, if the sqleseti API was issued in this connection. The current value of the CLIENT\_ACCTNG special register for this connection, unit of work, or activity.

This monitor element is synonymous to the **client\_acctng** monitor element. The **client\_acctng** monitor element is used for monitoring table functions and event monitors that write to unformatted tables, which were introduced in DB2 Version 9.7. The **tpmon\_acc\_str** monitor element is used for snapshot monitors and event monitors that write to tables, files, and pipes.

*Table 1266. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
DCS Application	dcs_appl	Basic

*Table 1267. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-
Deadlock	event_dlconn	-
Transaction	event_xact	-

### Usage

Use this element for problem determination and accounting purposes.

---

## tpmon\_client\_app - TP monitor client application name monitor element

Identifies the server transaction program performing the transaction, if the sqleseti API was issued in this connection. The current value of the CLIENT\_APPLNAME special register for this connection, unit of work, or activity.

This monitor element is synonymous to the **client\_applname** monitor element. The **client\_applname** monitor element is used for monitoring table functions and event monitors that write to unformatted tables, which were introduced in DB2 Version 9.7. The **tpmon\_client\_app** monitor element is used for snapshot monitors and event monitors that write to tables, files, and pipes.

*Table 1268. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
DCS Application	dcs_appl	Basic

Table 1269. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-
Deadlock	event_dlconn	-
Transaction	event_xact	-

## Usage

Use this element for problem determination and accounting purposes.

---

## tpmon\_client\_userid - TP monitor client user ID monitor element

The client user ID generated by a transaction manager and provided to the server, if the sqleseti API is used. The current value of the CLIENT\_USERID special register for this connection, unit of work, or activity.

This monitor element is synonymous to the **client\_userid** monitor element. The **client\_userid** monitor element is used for monitoring table functions and event monitors that write to unformatted tables, which were introduced in DB2 Version 9.7. The **tpmon\_client\_userid** monitor element is used for snapshot monitors and event monitors that write to tables, files, and pipes.

Table 1270. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
DCS Application	dcs_appl	Basic

Table 1271. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-
Deadlock	event_dlconn	-
Transaction	event_xact	-

## Usage

Use this element in application server or Transaction Processing monitor environments to identify the end-user for whom the transaction is being executed.

---

## tpmon\_client\_wkstn - TP monitor client workstation name monitor element

Identifies the client's system or workstation (for example CICS EITERMID), if the sqleseti API was issued in this connection. The current value of the CLIENT\_WRKSTNNAME special register for this connection, unit of work, or activity.

This monitor element is synonymous to the **client\_wrkstnname** monitor element. The **client\_wrkstnname** monitor element is used for monitoring table functions and event monitors that write to unformatted tables, which were introduced in

DB2 Version 9.7. The **tpmon\_client\_wkstn** monitor element is used for snapshot monitors and event monitors that write to tables, files, and pipes.

*Table 1272. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic
DCS Application	dcs_appl	Basic

*Table 1273. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity	-
Deadlock	event_dlconn	-
Transaction	event_xact	-

## Usage

Use this element to identify the user's machine by node ID, terminal ID, or similar identifiers.

---

## tq\_cur\_send\_spills - Current number of table queue buffers overflowed monitor element

The current number of table queue buffers residing in a temporary table.

*Table 1274. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

**Usage** An agent writing to a table queue may be sending rows to several readers. The writing agent will overflow buffers to a temporary table when the agent that it is currently sending rows to is not accepting rows and another agent requires rows in order to proceed. Overflowing to temporary table allows both the writer and the other readers to continue processing.

Rows that have been overflowed will be sent to the reading agent when it is ready to accept more rows.

If this number is high, and queries fail with sqlcode -968, and there are messages in db2diad.log indicating that you ran out of temporary space in the TEMP table space, then table queue overflows may be the cause. This could indicate a problem on another node (such as locking). You would investigate by taking snapshots on all the partitions for this query.

There are also cases, perhaps because of the way data is partitioned, where many buffers need to be overflowed for the query. In these cases you will need to add more disk to the temporary table space.

---

## tq\_id\_waiting\_on - Waited on node on a table queue

The agent that is waiting.

**Element identifier**

tq\_id\_waiting\_on

**Element type**  
information

Table 1275. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

**Usage** This can be used for troubleshooting.

---

## tq\_max\_send\_spills - Maximum number of table queue buffers overflows

Maximum number of table queue buffers overflowed to a temporary table.

**Element identifier**  
tq\_max\_send\_spills

**Element type**  
watermark

Table 1276. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

Table 1277. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_subsection	-

**Usage** Indicates the maximum number of table queue buffers that have been written to a temporary table.

---

## tq\_node\_waited\_for - Waited for node on a table queue

If the subsection status `ss_status` is *waiting to receive* or *waiting to send* and `tq_wait_for_any` is `FALSE`, then this is the number of the node that this agent is waiting for.

**Element identifier**  
tq\_node\_waited\_for

**Element type**  
information

Table 1278. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

**Usage** This can be used for troubleshooting. You may want to take an application snapshot on the node that the subsection is waiting for. For example, the application could be in a lock wait on that node.

---

## tq\_rows\_read - Number of Rows Read from table queues

Total number of rows read from table queues.



**Element identifier**  
tq\_rows\_read

**Element type**  
counter

*Table 1279. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

*Table 1280. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_subsection	-

**Usage** If monitoring does not indicate that this number is increasing, then processing progress is not taking place.

If there is significant differences in this number between nodes, then some nodes may be over utilized while others are being under utilized.

If this number is large, then there is a lot of data being shipped between nodes, suggest that optimization might improve the access plan.

---

## tq\_rows\_written - Number of rows written to table queues

Total number of rows written to table queues.

**Element identifier**  
tq\_rows\_written

**Element type**  
counter

*Table 1281. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

*Table 1282. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statements	event_subsection	-

**Usage** If monitoring does not indicate that this number is increasing, then processing progress is not taking place.

If there is significant differences in this number between nodes, then some nodes may be over utilized while others are being under utilized.

If this number is large, then there is a lot of data being shipped between nodes, suggest that optimization might improve the access plan.

---

## tq\_tot\_send\_spills - Total number of table queue buffers overflowed monitor element

Total number of table queue buffers overflowed to a temporary table.

Table 1283. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1284. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

Table 1285. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Statements	event_subsection	-

## Usage

Indicates the total number of table queue buffers that have been written to a temporary table. See the `tq_cur_send_spills` monitor element for more information.

---

## tq\_wait\_for\_any - Waiting for any node to send on a table queue

This flag is used to indicate that the subsection is blocked because it is waiting to receive rows from any node.

### Element identifier

tq\_wait\_for\_any

### Element type

information

Table 1286. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	subsection	Statement

**Usage** If `ss_status` indicates *waiting to receive data on a table queue* and this flag is TRUE, then the subsection is waiting to receive rows from any node. This generally indicates that the SQL statement has not processed to the point it can pass data to the waiting agent. For example, the writing agent may be performing a sort and will not write rows until the sort has completed. From the `db2expln` output, determine the subsection number associated with the `tablequeue` that the agent is waiting to receive rows from. You can then examine the status of that subsection by taking a snapshot on each node where it is executing.

---

## ts\_name - Table space being rolled forward monitor element

The name of the table space currently rolled forward.

**Element identifier**

ts\_name

**Element type**

information

Table 1287. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table Space	rollforward	Basic

**Usage** If a rollforward is in progress, this element identifies the table spaces involved.

---

## uid\_sql\_stmts - Update/Insert/Delete SQL Statements Executed

The number of SQL UPDATE, INSERT, and DELETE statements that were executed.

Table 1288. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 1289. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** You can use this element to determine the level of database activity at the application or database level.

You can also use the following formula to determine the ratio of UPDATE, INSERT and DELETE statements to the total number of statements:

`uid_sql_stmts`  
`/ (static_sql_stmts + dynamic_sql_stmts )`

This information can be useful for analyzing application activity and throughput.

---

## unread\_prefetch\_pages - Unread prefetch pages monitor element

Indicates the number of pages that the prefetcher read in that were never used.

*Table 1290. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE

*Table 1291. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Buffer Pool
Table Space	tablespace	Buffer Pool
Buffer Pool	bufferpool	Buffer Pool
Application	appl	Buffer Pool

For snapshot monitoring, this counter can be reset.

*Table 1292. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Tablespaces	event_tablespace	-
Connection	event_conn	-

### Usage

If this number is high, prefetchers are causing unnecessary I/O by reading pages into the buffer pool that will not be used.

---

## uow\_comp\_status - Unit of Work Completion Status

The status of the unit of work and how it stopped.

### Element identifier

`uow_comp_status`

### Element type

information

*Table 1293. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Unit of Work

Table 1293. Snapshot Monitoring Information (continued)

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcs_appl	Basic

Table 1294. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Transactions	event_xact	-

**Usage** You may use this element to determine if the unit of work ended due to a deadlock or abnormal termination. It may have been:

- Committed due to a commit statement
- Rolled back due to a rollback statement
- Rolled back due to a deadlock
- Rolled back due to an abnormal termination
- Committed at normal application termination.
- Unknown as a result of a FLUSH EVENT MONITOR command for which units of work were in progress.

**Note:** API users should refer to the header file (*sqlmon.h*) containing definitions of database system monitor constants.

---

## uow\_elapsed\_time - Most Recent Unit of Work Elapsed Time

The elapsed execution time of the most recently completed unit of work.

**Element identifier**

uow\_elapsed\_time

**Element type**

time

Table 1295. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Unit of Work, Timestamp
DCS Application	dcs_appl	Unit of Work, Timestamp

**Usage** Use this element as an indicator of the time it takes for units of work to complete.

---

## uow\_id - Unit of work ID monitor element

The unit of work identifier. The unit of work ID is unique within an application handle.

Table 1296. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics	Always collected

Table 1296. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details	Always collected

Table 1297. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-
Activities	event_activity	-
Activities	event_activitystmt	-
Activities	event_activityvals	-
Threshold violations	event_thresholdviolations	-

## Usage

Use this element in conjunction with other activity history elements for analysis of the behavior of an activity.

You can also use this element with the **activity\_id** and **appl\_id** monitor elements to uniquely identify an activity.

---

## uow\_lock\_wait\_time - Total time unit of work waited on locks monitor element

The total amount of elapsed time this unit of work has spent waiting for locks. The value is given in milliseconds.

### Element identifier

uow\_lock\_wait\_time

### Element type

counter

Table 1298. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Unit of Work

**Usage** This element can help you determine the severity of the resource contention problem.

---

## uow\_log\_space\_used - Unit of Work Log Space Used

The amount of log space (in bytes) used in the current unit of work of the monitored application.

### Element identifier

uow\_log\_space\_used

### Element type

gauge

Table 1299. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Unit of Work

Table 1300. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Transactions	event_xact	-

**Usage** You may use this element to understand the logging requirements at the unit of work level.

## uow\_start\_time - Unit of Work Start Timestamp

The date and time that the unit of work first required database resources.

Table 1301. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Unit of Work, Timestamp
DCS Application	dcs_appl	Unit of Work, Timestamp

Table 1302. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-

### Usage

This resource requirement occurs at the first SQL statement execution of that unit of work:

- For the first unit of work, it is the time of the first database request (SQL statement execution) after **conn\_complete\_time**.
- For subsequent units of work, it is the time of the first database request (SQL statement execution) after the previous COMMIT or ROLLBACK.

**Note:** The *SQL Reference* defines the boundaries of a unit of work as the COMMIT or ROLLBACK points.

The database system monitor excludes the time spent between the COMMIT/ROLLBACK and the next SQL statement from its definition of a unit of work. This measurement method reflects the time spent by the database manager in processing database requests, separate from time spent in application logic before the first SQL statement of that unit of work. The unit of work elapsed time does include the time spent running application logic between SQL statements within the unit of work.

You may use this element with the **uow\_stop\_time** monitor element to calculate the total elapsed time of the unit of work and with the **prev\_uow\_stop\_time** monitor element to calculate the time spent in the application between units of work.

You can use the **uow\_stop\_time** and the **prev\_uow\_stop\_time** monitor elements to calculate the elapsed time for the *SQL Reference* definition of a unit of work.

---

## uow\_status - Unit of Work Status

The status of the unit of work.

**Element identifier**  
uow\_status

**Element type**  
information

*Table 1303. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Transactions	event_xact	-

**Usage** You may use this element to determine the status of a unit of work. API users should refer to the sqlmon.h header file containing definitions of database system monitor constants.

---

## uow\_stop\_time - Unit of work stop timestamp monitor element

The date and time that the most recent unit of work completed, which occurs when database changes are committed or rolled back.

*Table 1304. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl	Unit of Work, Timestamp
DCS Application	dcs_appl	Unit of Work, Timestamp

*Table 1305. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-

### Usage

Use this element with the **prev\_uow\_stop\_time** monitor element to calculate the total elapsed time between COMMIT/ROLLBACK points, and with the **uow\_start\_time** monitor element to calculate the elapsed time of the latest unit of work.

The timestamp contents will be set as follows:

- When the application has completed a unit of work and has not yet started a new one (as defined in the **uow\_start\_time** monitor element), this element reports a valid, non-zero timestamp.
- When the application is currently executing a unit of work, this element reports zeros.
- When the application first connects to the database, this element is set to the value of the **conn\_complete\_time** monitor element

As a new unit of work is started, the contents of this element are moved to the **prev\_uow\_stop\_time** monitor element.



---

## uow\_total\_time\_top - UOW total time top monitor element

Reserved for future use.

Table 1306. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wlstats	-
Statistics	event_scstats	-

### Usage

Reserved for future use.

---

## update\_sql\_stmts - Updates

This element contains a count of the total number of times the federated server has issued an UPDATE statement to this data source on behalf of any application from the start of the federated server instance, or the last reset of the database monitor counters.

Table 1307. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Basic
Application	appl_remote	Basic

For snapshot monitoring, this counter can be reset.

**Usage** Use this element to determine the level of database activity directed against this data source by the federated server or an application.

You can also use this element to determine the percentage of write activity against this data source by the federated server or an application, with the following formula:

$$\text{write\_activity} = \frac{(\text{INSERT statements} + \text{UPDATE statements} + \text{DELETE statements})}{(\text{SELECT statements} + \text{INSERT statements} + \text{UPDATE statements} + \text{DELETE statements})}$$

---

## update\_time - Update Response Time

This element contains the aggregate amount of time, in milliseconds, that it has taken this data source to respond to UPDATES from all applications or a single application running on this federated server instance from the start of the federated server instance, or the last reset of the database monitor counters.

Table 1308. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase_remote	Timestamp
Application	appl_remote	Timestamp

For snapshot monitoring, this counter can be reset.

The response time is measured as the difference in time between the time the federated server submits an UPDATE statement to the data source, and the time the data source responds to the federated server, indicating the UPDATE has been processed.

**Usage** Use this element to determine how much actual time transpires while waiting for UPDATES to this data source to be processed. This information can be useful for capacity planning and tuning.

## user\_cpu\_time - User CPU Time

The total *user* CPU time (in seconds and microseconds) used by the database manager agent process, the unit of work, or the statement.

When either the statement monitor switch or the timestamp switch is not turned on, this element is not collected and -1 is written instead.

*Table 1309. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Connection	event_conn	-
Transactions	event_xact	-
Statements	event_stmt	-
Activities	event_activity	-

**Usage** This element along with the other related CPU-time elements can help you understand the level of activity within an application, and can help you identify applications that could benefit from additional tuning.

**Note:** If this information is not available for your operating system, this element will be set to 0.

## utility\_dbname - Database Operated on by Utility

The database operated on by the utility.

*Table 1310. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	utility_info	Basic

## utility\_description - Utility Description

A brief description of the work a utility is performing. For example, a rebalance invocation may contain "Tablespace ID: 2" representing that this rebalancer is working on table space with ID 2. The format of this field is dependent on the class of utility and is subject to change between releases.

*Table 1311. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	utility_info	Basic

---

## utility\_id - Utility ID

The unique identifier corresponding to the utility invocation.

*Table 1312. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	Always collected

*Table 1313. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	utility_info	Basic

---

## utility\_invoker\_type - Utility Invoker Type

This element describes how a utility was invoked.

*Table 1314. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	utility_info	Basic

**Usage** Use this element to determine how a utility was invoked. For example, you can use it to determine whether a utility was invoked automatically by DB2 or by a user. The values for this element, listed as follows, are defined in sqlmon.h.

API Constant	Utility
SQLM_UTILITY_INVOKER_USER	Utility was invoked by user
SQLM_UTILITY_INVOKER_AUTO	Utility was invoked automatically by DB2

---

## utility\_priority - Utility Priority

Utility priority specifies the amount of relative importance of a throttled utility with respect to its throttled peers. A priority of 0 implies that a utility is executing unthrottled. Non-zero priorities must fall in the range of 1-100, with 100 representing the highest priority and 1 representing the lowest.

*Table 1315. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	utility_info	Basic

---

## utility\_start\_time - Utility Start Time

The date and time when the current utility was originally invoked.

Table 1316. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	utility_info	Basic

## utility\_state - Utility State

This element describes the state of a utility.

**Element identifier**

utility\_state

**Element type**

information

Table 1317. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	utility_info	Basic

**Usage** Use this element to determine the state of an active utility. The values for this field, listed as follows, are defined in sqlmon.h.

API Constant	Description
SQLM_UTILITY_STATE_EXECUTE	Utility is executing
SQLM_UTILITY_STATE_WAIT	Utility is waiting for an event to occur before resuming progress
SQLM_UTILITY_STATE_ERROR	Utility has encountered an error

## valid - Section validity indicator monitor element

Indicates whether the dynamic SQL statement section is valid. For static SQL statements, the value of this monitor element is always Y.

Table 1318. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

**Usage**

Valid values for this monitor element are Y and N. An invalid section will be implicitly prepared by the system when next used.

## utility\_type - Utility Type

The class of utility.

Table 1319. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database Manager	utility_info	Basic

## Usage

The values for this element can be any of the constants defined in `sqlmon.h` with names beginning "SQLM\_UTILITY\_".

---

### vectored\_ios - Number of vectored I/O requests monitor element

The number of vectored I/O requests. More specifically, the number of times DB2 performs sequential prefetching of pages into the page area of the buffer pool.

*Table 1320. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_BUFFERPOOL table function - Get buffer pool metrics	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE table function - Get table space metrics	DATA OBJECT METRICS BASE
MON_GET_CONTAINER table function - Get table space container metrics	DATA OBJECT METRICS BASE

*Table 1321. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Buffer Pool	bufferpool	Buffer Pool

## Usage

Use this element to determine how often vectored I/O is being done. The number of vectored I/O requests is monitored only during sequential prefetching.

---

### version - Version of Monitor Data

The version of the database manager that produced the event monitor data stream.

*Table 1322. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Event Log Header	event_log_header	-

## Usage

The data structures used by the event monitor may change between releases of the database manager. As a result, your monitor applications should check the version of the data stream to determine if they can process the data they will be receiving.

For this release, this element is set to the API constant `SQLM_DBMON_VERSION9_5`.

---

### wlm\_queue\_assignments\_total - Workload manager total queue assignments monitor element

The number of times that activities have been queued by a WLM threshold.

Table 1323. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

## wlm\_queue\_time\_total - Workload manager total queue time monitor element

The time spent waiting on a WLM queuing threshold. This value is given in milliseconds.

Table 1324. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_CONNECTION table function - Get connection metrics	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS table function - Get detailed connection metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS table function - Get service subclass metrics	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS table function - Get detailed service subclass metrics (reported in DETAILS XML document)	REQUEST METRICS BASE

Table 1324. Table Function Monitoring Information (continued)

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_WORKLOAD table function - Get workload metrics	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics (reported in DETAILS XML document)	REQUEST METRICS BASE
MON_GET_ACTIVITY_DETAILS table function - Get complete activity details (reported in DETAILS XML document)	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT table function - Get SQL statement activity metrics in the package cache	ACTIVITY METRICS BASE

## wlo\_completed\_total - Workload occurrences completed total monitor element

The number of workload occurrences to complete since last reset.

Table 1325. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wlstats	-

### Usage

Use this element to determine how many occurrences of a given workload are driving work into the system.

## work\_action\_set\_id - Work action set ID monitor element

The ID of the work action set to which this statistics record applies.

Table 1326. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_histogrambin	-
Statistics	event_wcstats	-

### Usage

Use this element in conjunction with other activity history elements for analysis of the behavior of an activity or with other statistics elements for analysis of a work class.

---

## work\_action\_set\_name - Work action set name monitor element

The name of the work action set to which the statistics shown as part of this event are associated.

*Table 1327. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_qstats	-
Statistics	event_wcstats	-

### Usage

Use this element along with the **work\_class\_name** element to uniquely identify the work class whose statistics are being shown in this record or to uniquely identify the work class which is the domain of the threshold queue whose statistics are shown in this record.

---

## work\_class\_id - Work class ID monitor element

The identifier of the work class to which this statistics record applies.

*Table 1328. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_wcstats	-
Statistics	event_histogrambin	-

### Usage

Use this element in conjunction with other statistics elements for analysis of a work class.

---

## work\_class\_name - Work class name monitor element

The name of the work class to which the statistics shown as part of this event are associated.

*Table 1329. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Statistics	event_qstats	-
Statistics	event_wcstats	-

### Usage

Use this element along with the **work\_action\_set\_name** element to uniquely identify the work class whose statistics are being shown in this record or to uniquely identify the work class which is the domain of the threshold queue whose statistics are shown in this record.

---

## workload\_id - Workload ID monitor element

An integer that uniquely identifies a workload.



Table 1330. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_WORKLOAD table function - Get workload metrics	Always collected
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics	Always collected

Table 1331. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Application	appl_info	Basic

Table 1332. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Statistics	event_wlstats	-
Statistics	event_histogrambin	-
Activities	event_activity	-

## Usage

Use this ID to uniquely identify the workload to which this activity, application, histogram bin, or workload statistics record belongs.

---

## workload\_name - Workload name monitor element

Name of the workload.

Table 1333. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics	Always collected
MON_GET_WORKLOAD table function - Get workload metrics	Always collected
MON_GET_WORKLOAD_DETAILS table function - Get detailed workload metrics	Always collected

Table 1334. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Locking	-	-
Unit of work	-	-
Activities	event_activity (reported in the details_xml document)	ACTIVITY METRICS BASE
Statistics	event_scstats (reported in the details_xml document)	REQUEST METRICS BASE
Statistics	event_wlstats (reported in the details_xml document)	REQUEST METRICS BASE
Unit of work	Reported in the in the system_metrics document.	-
Statistics	event_wlstats	-

## Usage

In the statistics event monitor and workload table functions, the workload name identifies the workload for which statistics or metrics are being collected and reported. In the unit of work event monitor and unit of work table functions, the workload name identifies the workload that the unit of work was associated with.

Use the workload name to identify units of work or sets of information that apply to a particular workload of interest.

---

## workload\_occurrence\_id - Workload occurrence identifier monitor element

The ID of the workload occurrence to which this activity belongs.

Table 1335. Table Function Monitoring Information

Table Function	Monitor Element Collection Level
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics	Always collected

Table 1336. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Unit of work	-	-
Activities	event_activity	-

## Usage

Use this to identify the workload occurrence that submitted the activity.

---

## workload\_occurrence\_state - Workload occurrence state monitor element

The state of the workload occurrence.

*Table 1337. Table Function Monitoring Information*

Table Function	Monitor Element Collection Command and Level
MON_GET_UNIT_OF_WORK table function - Get unit of work metrics	Always collected
MON_GET_UNIT_OF_WORK_DETAILS table function - Get detailed unit of work metrics (reported in DETAILS XML document)	Always collected

### Usage

Possible values include:

#### DECOUPLED

Workload occurrence does not have a coordinator agent assigned (concentrator case).

#### DISCONNECTPEND

Workload occurrence is disconnecting from the database.

#### FORCED

Workload occurrence has been forced.

#### INTERRUPTED

Workload occurrence has been interrupted.

#### QUEUED

Workload occurrence coordinator agent is queued by Query Patroller or a workload management queuing threshold. In a database partitioning feature (DPF) environment, this state may indicate that the coordinator agent has made an RPC to the catalog partition to obtain threshold tickets and has not yet received a response.

#### TRANSIENT

Workload occurrence has not yet been mapped to a service superclass.

#### UOWEXEC

Workload occurrence is processing a request.

#### UOWWAIT

Workload occurrence is waiting for a request from the client.

---

## x\_lock\_escals - Exclusive Lock Escalations

The number of times that locks have been escalated from several row locks to one exclusive table lock, or the number of times an exclusive lock on a row caused the table lock to become an exclusive lock.

#### Element identifier

x\_lock\_escals

#### Element type

counter

Table 1338. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

Table 1339. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-
Transactions	event_xact	-

**Usage** Other applications cannot access data held by an exclusive lock; therefore it is important to track exclusive locks since they can impact the concurrency of your data.

A lock is escalated when the total number of locks held by an application reaches the maximum amount of lock list space available to the application. The amount of lock list space available is determined by the *locklist* and *maxlocks* configuration parameters.

When an application reaches the maximum number of locks allowed and there are no more locks to escalate, it will then use space in the lock list allocated for other applications. When the entire lock list is full, an error occurs.

See *lock\_escals* for possible causes and resolutions to excessive exclusive lock escalations.

An application may be using exclusive locks when share locks are sufficient. Although share locks may not reduce the total number of lock escalations share lock escalations may be preferable to exclusive lock escalations.

---

## xda\_object\_pages - XDA Object Pages

The number of disk pages consumed by XML storage object (XDA) data.

**Element identifier**

xda\_object\_pages

**Element type**

information

Table 1340. Snapshot Monitoring Information

Snapshot Level	Logical Data Grouping	Monitor Switch
Table	table	Basic

Table 1341. Event Monitoring Information

Event Type	Logical Data Grouping	Monitor Switch
Tables	event_table	-

**Usage** This element provides a mechanism for viewing the actual amount of

space consumed by XML storage object (XDA) data in a particular table. This element can be used in conjunction with a table event monitor to track the rate of XML storage object data growth over time.

---

## xid - Transaction ID

A unique transaction identifier (across all databases) generated by a transaction manager in a two-phase commit transaction.

*Table 1342. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
DCS Application	dcx_appl	Unit of Work

**Usage** This identifier can be used to correlate the transaction generated by the transaction manager with the transactions executed against multiple databases. It can be used to help diagnose transaction manager problems by tying database transactions involving a two-phase commit protocol with the transactions originated by the transaction manager.

---

## xquery\_stmts - XQuery Statements Attempted

The number of XQuery statements executed for an application or database.

**Element identifier**

xquery\_stmts

**Element type**

counter

*Table 1343. Snapshot Monitoring Information*

Snapshot Level	Logical Data Grouping	Monitor Switch
Database	dbase	Basic
Application	appl	Basic

For snapshot monitoring, this counter can be reset.

*Table 1344. Event Monitoring Information*

Event Type	Logical Data Grouping	Monitor Switch
Database	event_db	-
Connection	event_conn	-

**Usage** You can use this element to gauge the activity of native XQuery language requests. This does not include embedded XQuery language requests such as xmlquery, xmltable, or xmlexist.

---

## Part 3. Appendixes



---

## Appendix A. Overview of the DB2 technical information

DB2 technical information is available through the following tools and methods:

- DB2 Information Center
  - Topics (Task, concept and reference topics)
  - Help for DB2 tools
  - Sample programs
  - Tutorials
- DB2 books
  - PDF files (downloadable)
  - PDF files (from the DB2 PDF DVD)
  - printed books
- Command line help
  - Command help
  - Message help

**Note:** The DB2 Information Center topics are updated more frequently than either the PDF or the hardcopy books. To get the most current information, install the documentation updates as they become available, or refer to the DB2 Information Center at [ibm.com](http://ibm.com).

You can access additional DB2 technical information such as technotes, white papers, and IBM Redbooks® publications online at [ibm.com](http://ibm.com). Access the DB2 Information Management software library site at <http://www.ibm.com/software/data/sw-library/>.

### Documentation feedback

We value your feedback on the DB2 documentation. If you have suggestions for how to improve the DB2 documentation, send an e-mail to [db2docs@ca.ibm.com](mailto:db2docs@ca.ibm.com). The DB2 documentation team reads all of your feedback, but cannot respond to you directly. Provide specific examples wherever possible so that we can better understand your concerns. If you are providing feedback on a specific topic or help file, include the topic title and URL.

Do not use this e-mail address to contact DB2 Customer Support. If you have a DB2 technical issue that the documentation does not resolve, contact your local IBM service center for assistance.

---

## DB2 technical library in hardcopy or PDF format

The following tables describe the DB2 library available from the IBM Publications Center at [www.ibm.com/shop/publications/order](http://www.ibm.com/shop/publications/order). English and translated DB2 Version 9.7 manuals in PDF format can be downloaded from [www.ibm.com/support/docview.wss?rs=71&uid=swg2700947](http://www.ibm.com/support/docview.wss?rs=71&uid=swg2700947).

Although the tables identify books available in print, the books might not be available in your country or region.



The form number increases each time a manual is updated. Ensure that you are reading the most recent version of the manuals, as listed below.

**Note:** The *DB2 Information Center* is updated more frequently than either the PDF or the hard-copy books.

*Table 1345. DB2 technical information*

<b>Name</b>	<b>Form Number</b>	<b>Available in print</b>	<b>Last updated</b>
<i>Administrative API Reference</i>	SC27-2435-00	Yes	August, 2009
<i>Administrative Routines and Views</i>	SC27-2436-00	No	August, 2009
<i>Call Level Interface Guide and Reference, Volume 1</i>	SC27-2437-00	Yes	August, 2009
<i>Call Level Interface Guide and Reference, Volume 2</i>	SC27-2438-00	Yes	August, 2009
<i>Command Reference</i>	SC27-2439-00	Yes	August, 2009
<i>Data Movement Utilities Guide and Reference</i>	SC27-2440-00	Yes	August, 2009
<i>Data Recovery and High Availability Guide and Reference</i>	SC27-2441-00	Yes	August, 2009
<i>Database Administration Concepts and Configuration Reference</i>	SC27-2442-00	Yes	August, 2009
<i>Database Monitoring Guide and Reference</i>	SC27-2458-00	Yes	August, 2009
<i>Database Security Guide</i>	SC27-2443-00	Yes	August, 2009
<i>DB2 Text Search Guide</i>	SC27-2459-00	Yes	August, 2009
<i>Developing ADO.NET and OLE DB Applications</i>	SC27-2444-00	Yes	August, 2009
<i>Developing Embedded SQL Applications</i>	SC27-2445-00	Yes	August, 2009
<i>Developing Java Applications</i>	SC27-2446-00	Yes	August, 2009
<i>Developing Perl, PHP, Python, and Ruby on Rails Applications</i>	SC27-2447-00	No	August, 2009
<i>Developing User-defined Routines (SQL and External)</i>	SC27-2448-00	Yes	August, 2009
<i>Getting Started with Database Application Development</i>	GI11-9410-00	Yes	August, 2009
<i>Getting Started with DB2 Installation and Administration on Linux and Windows</i>	GI11-9411-00	Yes	August, 2009

*Table 1345. DB2 technical information (continued)*

<b>Name</b>	<b>Form Number</b>	<b>Available in print</b>	<b>Last updated</b>
<i>Globalization Guide</i>	SC27-2449-00	Yes	August, 2009
<i>Installing DB2 Servers</i>	GC27-2455-00	Yes	August, 2009
<i>Installing IBM Data Server Clients</i>	GC27-2454-00	No	August, 2009
<i>Message Reference Volume 1</i>	SC27-2450-00	No	August, 2009
<i>Message Reference Volume 2</i>	SC27-2451-00	No	August, 2009
<i>Net Search Extender Administration and User's Guide</i>	SC27-2469-00	No	August, 2009
<i>Partitioning and Clustering Guide</i>	SC27-2453-00	Yes	August, 2009
<i>pureXML Guide</i>	SC27-2465-00	Yes	August, 2009
<i>Query Patroller Administration and User's Guide</i>	SC27-2467-00	No	August, 2009
<i>Spatial Extender and Geodetic Data Management Feature User's Guide and Reference</i>	SC27-2468-00	No	August, 2009
<i>SQL Procedural Languages: Application Enablement and Support</i>	SC27-2470-00	Yes	August, 2009
<i>SQL Reference, Volume 1</i>	SC27-2456-00	Yes	August, 2009
<i>SQL Reference, Volume 2</i>	SC27-2457-00	Yes	August, 2009
<i>Troubleshooting and Tuning Database Performance</i>	SC27-2461-00	Yes	August, 2009
<i>Upgrading to DB2 Version 9.7</i>	SC27-2452-00	Yes	August, 2009
<i>Visual Explain Tutorial</i>	SC27-2462-00	No	August, 2009
<i>What's New for DB2 Version 9.7</i>	SC27-2463-00	Yes	August, 2009
<i>Workload Manager Guide and Reference</i>	SC27-2464-00	Yes	August, 2009
<i>XQuery Reference</i>	SC27-2466-00	No	August, 2009

*Table 1346. DB2 Connect-specific technical information*

<b>Name</b>	<b>Form Number</b>	<b>Available in print</b>	<b>Last updated</b>
<i>Installing and Configuring DB2 Connect Personal Edition</i>	SC27-2432-00	Yes	August, 2009
<i>Installing and Configuring DB2 Connect Servers</i>	SC27-2433-00	Yes	August, 2009

Table 1346. DB2 Connect-specific technical information (continued)

Name	Form Number	Available in print	Last updated
DB2 Connect User's Guide	SC27-2434-00	Yes	August, 2009

Table 1347. Information Integration technical information

Name	Form Number	Available in print	Last updated
Information Integration: Administration Guide for Federated Systems	SC19-1020-02	Yes	August, 2009
Information Integration: ASNCLP Program Reference for Replication and Event Publishing	SC19-1018-04	Yes	August, 2009
Information Integration: Configuration Guide for Federated Data Sources	SC19-1034-02	No	August, 2009
Information Integration: SQL Replication Guide and Reference	SC19-1030-02	Yes	August, 2009
Information Integration: Introduction to Replication and Event Publishing	GC19-1028-02	Yes	August, 2009

## Ordering printed DB2 books

If you require printed DB2 books, you can buy them online in many but not all countries or regions. You can always order printed DB2 books from your local IBM representative. Keep in mind that some softcopy books on the *DB2 PDF Documentation DVD* are unavailable in print. For example, neither volume of the *DB2 Message Reference* is available as a printed book.

Printed versions of many of the DB2 books available on the *DB2 PDF Documentation DVD* can be ordered for a fee from IBM. Depending on where you are placing your order from, you may be able to order books online, from the IBM Publications Center. If online ordering is not available in your country or region, you can always order printed DB2 books from your local IBM representative. Note that not all books on the *DB2 PDF Documentation DVD* are available in print.

**Note:** The most up-to-date and complete DB2 documentation is maintained in the DB2 Information Center at <http://publib.boulder.ibm.com/infocenter/db2luw/v9r7>.

To order printed DB2 books:

- To find out whether you can order printed DB2 books online in your country or region, check the IBM Publications Center at <http://www.ibm.com/shop/publications/order>. You must select a country, region, or language to access publication ordering information and then follow the ordering instructions for your location.
- To order printed DB2 books from your local IBM representative:

1. Locate the contact information for your local representative from one of the following Web sites:
  - The IBM directory of world wide contacts at [www.ibm.com/planetwide](http://www.ibm.com/planetwide)
  - The IBM Publications Web site at <http://www.ibm.com/shop/publications/order>. You will need to select your country, region, or language to the access appropriate publications home page for your location. From this page, follow the "About this site" link.
2. When you call, specify that you want to order a DB2 publication.
3. Provide your representative with the titles and form numbers of the books that you want to order. For titles and form numbers, see "DB2 technical library in hardcopy or PDF format" on page 753.

---

## Displaying SQL state help from the command line processor

DB2 products return an SQLSTATE value for conditions that can be the result of an SQL statement. SQLSTATE help explains the meanings of SQL states and SQL state class codes.

To start SQL state help, open the command line processor and enter:

```
? sqlstate or ? class code
```

where *sqlstate* represents a valid five-digit SQL state and *class code* represents the first two digits of the SQL state.

For example, ? 08003 displays help for the 08003 SQL state, and ? 08 displays help for the 08 class code.

---

## Accessing different versions of the DB2 Information Center

For DB2 Version 9.7 topics, the DB2 Information Center URL is <http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/>

For DB2 Version 9.5 topics, the DB2 Information Center URL is <http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/>

For DB2 Version 9 topics, the DB2 Information Center URL is <http://publib.boulder.ibm.com/infocenter/db2luw/v9/>

For DB2 Version 8 topics, go to the Version 8 Information Center URL at: <http://publib.boulder.ibm.com/infocenter/db2luw/v8/>

---

## Displaying topics in your preferred language in the DB2 Information Center

The DB2 Information Center attempts to display topics in the language specified in your browser preferences. If a topic has not been translated into your preferred language, the DB2 Information Center displays the topic in English.

- To display topics in your preferred language in the Internet Explorer browser:
  1. In Internet Explorer, click the **Tools** —> **Internet Options** —> **Languages...** button. The Language Preferences window opens.
  2. Ensure your preferred language is specified as the first entry in the list of languages.
    - To add a new language to the list, click the **Add...** button.

**Note:** Adding a language does not guarantee that the computer has the fonts required to display the topics in the preferred language.

- To move a language to the top of the list, select the language and click the **Move Up** button until the language is first in the list of languages.
- 3. Clear the browser cache and then refresh the page to display the DB2 Information Center in your preferred language.
- To display topics in your preferred language in a Firefox or Mozilla browser:
  1. Select the button in the **Languages** section of the **Tools** —> **Options** —> **Advanced** dialog. The Languages panel is displayed in the Preferences window.
  2. Ensure your preferred language is specified as the first entry in the list of languages.
    - To add a new language to the list, click the **Add...** button to select a language from the Add Languages window.
    - To move a language to the top of the list, select the language and click the **Move Up** button until the language is first in the list of languages.
  3. Clear the browser cache and then refresh the page to display the DB2 Information Center in your preferred language.

On some browser and operating system combinations, you must also change the regional settings of your operating system to the locale and language of your choice.

---

## Updating the DB2 Information Center installed on your computer or intranet server

A locally installed DB2 Information Center must be updated periodically.

### Before you begin

A DB2 Version 9.7 Information Center must already be installed. For details, see the “Installing the DB2 Information Center using the DB2 Setup wizard” topic in *Installing DB2 Servers*. All prerequisites and restrictions that applied to installing the Information Center also apply to updating the Information Center.

### About this task

An existing DB2 Information Center can be updated automatically or manually:

- Automatic updates - updates existing Information Center features and languages. An additional benefit of automatic updates is that the Information Center is unavailable for a minimal period of time during the update. In addition, automatic updates can be set to run as part of other batch jobs that run periodically.
- Manual updates - should be used when you want to add features or languages during the update process. For example, a local Information Center was originally installed with both English and French languages, and now you want to also install the German language; a manual update will install German, as well as, update the existing Information Center features and languages. However, a manual update requires you to manually stop, update, and restart the Information Center. The Information Center is unavailable during the entire update process.

### Procedure

This topic details the process for automatic updates. For manual update instructions, see the “Manually updating the DB2 Information Center installed on your computer or intranet server” topic.

To automatically update the DB2 Information Center installed on your computer or intranet server:

1. On Linux operating systems,
  - a. Navigate to the path where the Information Center is installed. By default, the DB2 Information Center is installed in the /opt/ibm/db2ic/V9.7 directory.
  - b. Navigate from the installation directory to the doc/bin directory.
  - c. Run the ic-update script:

```
ic-update
```
2. On Windows operating systems,
  - a. Open a command window.
  - b. Navigate to the path where the Information Center is installed. By default, the DB2 Information Center is installed in the <Program Files>\IBM\DB2 Information Center\Version 9.7 directory, where <Program Files> represents the location of the Program Files directory.
  - c. Navigate from the installation directory to the doc\bin directory.
  - d. Run the ic-update.bat file:

```
ic-update.bat
```

## Results

The DB2 Information Center restarts automatically. If updates were available, the Information Center displays the new and updated topics. If Information Center updates were not available, a message is added to the log. The log file is located in doc\eclipse\configuration directory. The log file name is a randomly generated number. For example, 1239053440785.log.

---

## Manually updating the DB2 Information Center installed on your computer or intranet server

If you have installed the DB2 Information Center locally, you can obtain and install documentation updates from IBM.

Updating your locally-installed DB2 Information Center manually requires that you:

1. Stop the DB2 Information Center on your computer, and restart the Information Center in stand-alone mode. Running the Information Center in stand-alone mode prevents other users on your network from accessing the Information Center, and allows you to apply updates. The Workstation version of the DB2 Information Center always runs in stand-alone mode. .
2. Use the Update feature to see what updates are available. If there are updates that you must install, you can use the Update feature to obtain and install them

**Note:** If your environment requires installing the DB2 Information Center updates on a machine that is not connected to the internet, mirror the update site to a local file system using a machine that is connected to the internet and has the DB2 Information Center installed. If many users on your network will be installing the documentation updates, you can reduce the time required for

individuals to perform the updates by also mirroring the update site locally and creating a proxy for the update site.

If update packages are available, use the Update feature to get the packages. However, the Update feature is only available in stand-alone mode.

3. Stop the stand-alone Information Center, and restart the DB2 Information Center on your computer.

**Note:** On Windows 2008, Windows Vista (and higher), the commands listed later in this section must be run as an administrator. To open a command prompt or graphical tool with full administrator privileges, right-click the shortcut and then select **Run as administrator**.

To update the DB2 Information Center installed on your computer or intranet server:

1. Stop the DB2 Information Center.
  - On Windows, click **Start** → **Control Panel** → **Administrative Tools** → **Services**. Then right-click **DB2 Information Center** service and select **Stop**.
  - On Linux, enter the following command:  


```
/etc/init.d/db2icdv97 stop
```

2. Start the Information Center in stand-alone mode.
  - On Windows:
    - a. Open a command window.
    - b. Navigate to the path where the Information Center is installed. By default, the DB2 Information Center is installed in the <Program Files>\IBM\DB2 Information Center\Version 9.7 directory, where <Program Files> represents the location of the Program Files directory.
    - c. Navigate from the installation directory to the doc\bin directory.
    - d. Run the help\_start.bat file:  

```
help_start.bat
```
  - On Linux:
    - a. Navigate to the path where the Information Center is installed. By default, the DB2 Information Center is installed in the /opt/ibm/db2ic/V9.7 directory.
    - b. Navigate from the installation directory to the doc/bin directory.
    - c. Run the help\_start script:  

```
help_start
```

The systems default Web browser opens to display the stand-alone Information Center.

3. Click the **Update** button (). (JavaScript™ must be enabled in your browser.) On the right panel of the Information Center, click **Find Updates**. A list of updates for existing documentation displays.
4. To initiate the installation process, check the selections you want to install, then click **Install Updates**.
5. After the installation process has completed, click **Finish**.
6. Stop the stand-alone Information Center:
  - On Windows, navigate to the installation directory's doc\bin directory, and run the help\_end.bat file:  

```
help_end.bat
```



**Note:** The help\_end batch file contains the commands required to safely stop the processes that were started with the help\_start batch file. Do not use Ctrl-C or any other method to stop help\_start.bat.

- On Linux, navigate to the installation directory's doc/bin directory, and run the help\_end script:  
help\_end

**Note:** The help\_end script contains the commands required to safely stop the processes that were started with the help\_start script. Do not use any other method to stop the help\_start script.

7. Restart the DB2 Information Center.

- On Windows, click **Start** → **Control Panel** → **Administrative Tools** → **Services**. Then right-click **DB2 Information Center** service and select **Start**.
- On Linux, enter the following command:  
/etc/init.d/db2icdv97 start

The updated DB2 Information Center displays the new and updated topics.

---

## DB2 tutorials

The DB2 tutorials help you learn about various aspects of DB2 products. Lessons provide step-by-step instructions.

### Before you begin

You can view the XHTML version of the tutorial from the Information Center at <http://publib.boulder.ibm.com/infocenter/db2help/>.

Some lessons use sample data or code. See the tutorial for a description of any prerequisites for its specific tasks.

### DB2 tutorials

To view the tutorial, click the title.

#### **“pureXML®” in *pureXML Guide***

Set up a DB2 database to store XML data and to perform basic operations with the native XML data store.

#### **“Visual Explain” in *Visual Explain Tutorial***

Analyze, optimize, and tune SQL statements for better performance using Visual Explain.

---

## DB2 troubleshooting information

A wide variety of troubleshooting and problem determination information is available to assist you in using DB2 database products.

### DB2 documentation

Troubleshooting information can be found in the *DB2 Troubleshooting Guide* or the Database fundamentals section of the *DB2 Information Center*. There you will find information about how to isolate and identify problems using DB2 diagnostic tools and utilities, solutions to some of the most common problems, and other advice on how to solve problems you might encounter with your DB2 database products.



### DB2 Technical Support Web site

Refer to the DB2 Technical Support Web site if you are experiencing problems and want help finding possible causes and solutions. The Technical Support site has links to the latest DB2 publications, TechNotes, Authorized Program Analysis Reports (APARs or bug fixes), fix packs, and other resources. You can search through this knowledge base to find possible solutions to your problems.

Access the DB2 Technical Support Web site at [http://www.ibm.com/software/data/db2/support/db2\\_9/](http://www.ibm.com/software/data/db2/support/db2_9/)

---

## Terms and Conditions

Permissions for the use of these publications is granted subject to the following terms and conditions.

**Personal use:** You may reproduce these Publications for your personal, non commercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these Publications, or any portion thereof, without the express consent of IBM.

**Commercial use:** You may reproduce, distribute and display these Publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these Publications, or reproduce, distribute or display these Publications or any portion thereof outside your enterprise, without the express consent of IBM.

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the Publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the Publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

---

## Appendix B. Notices

This information was developed for products and services offered in the U.S.A. Information about non-IBM products is based on information available at the time of first publication of this document and is subject to change.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing  
IBM Corporation  
North Castle Drive  
Armonk, NY 10504-1785  
U.S.A.

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing  
Legal and Intellectual Property Law  
IBM Japan, Ltd.  
3-2-12, Roppongi, Minato-ku, Tokyo 106-8711 Japan

**The following paragraph does not apply to the United Kingdom or any other country/region where such provisions are inconsistent with local law:**

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions; therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web

sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information that has been exchanged, should contact:

IBM Canada Limited  
Office of the Lab Director  
8200 Warden Avenue  
Markham, Ontario  
L6G 1C7  
CANADA

Such information may be available, subject to appropriate terms and conditions, including, in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems, and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements, or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information may contain examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious, and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

#### COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application

programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work must include a copyright notice as follows:

© (*your company name*) (*year*). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. *\_enter the year or years\_*. All rights reserved.

## Trademarks

IBM, the IBM logo, and [ibm.com](http://ibm.com)<sup>®</sup> are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at [www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml).

The following terms are trademarks or registered trademarks of other companies

- Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
- Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.
- UNIX is a registered trademark of The Open Group in the United States and other countries.
- Intel, Intel logo, Intel Inside<sup>®</sup>, Intel Inside logo, Intel<sup>®</sup> Centrino<sup>®</sup>, Intel Centrino logo, Celeron<sup>®</sup>, Intel<sup>®</sup> Xeon<sup>®</sup>, Intel SpeedStep<sup>®</sup>, Itanium<sup>®</sup>, and Pentium<sup>®</sup> are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.
- Microsoft, Windows, Windows NT<sup>®</sup>, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.



---

# Index

## Special characters

.db2toprc configuration file 108

## A

activation time

last\_wlm\_reset monitor element 475

activities

monitor elements

act\_aborted\_total 302

act\_completed\_total 303

act\_rejected\_total 305

act\_total 307

activity\_collected 309

activity\_id 309

activity\_secondary\_id 310

activity\_state 310

activity\_type 311

coord\_act\_aborted\_total 376

coord\_act\_completed\_total 376

coord\_act\_rejected\_total 382

parent\_activity\_id 538

Activity Monitor

overview 190

setup 196

ACTIVITYTOTALTIME activity threshold

monitor elements

activitytotaltime\_threshold\_id 311

activitytotaltime\_threshold\_value 312

activitytotaltime\_threshold\_violated 312

administrative views

APPL\_PERFORMANCE scenario 193

BP\_HITRATIO scenario 195

BP\_READ\_IO scenario 195

BP\_WRITE\_IO scenario 195

LONG\_RUNNING\_SQL scenario 193

QUERY\_PREP\_COST scenario 193

TOP\_DYNAMIC\_SQL

scenario 193

agents

monitor elements

agent\_id 313

agent\_id\_holding\_lock 314

agent\_pid 315

agent\_status 315

agent\_sys\_cpu\_time 315

agent\_usr\_cpu\_time 316

agent\_wait\_time 317

agent\_waits\_total 318

agents\_created\_empty\_pool 319

agents\_from\_pool 319

agents\_registered 320

agents\_registered\_top 320

agents\_stolen 320

agents\_top 321

agents\_waiting\_on\_token 321

agents\_waiting\_top 321

appl\_priority 329

associated\_agents\_top 335

coord\_agent\_pid 382

agents (*continued*)

monitor elements (*continued*)

coord\_agents\_top 383

idle\_agents 458

locks\_waiting 495

max\_agent\_overflows 502

num\_agents 518

num\_assoc\_agents 519

priv\_workspace\_size\_top element 599

quiescer\_agent\_id 605

rolled\_back\_agent\_id 620

alert actions

health indicators

states 158

alert thresholds

configuration

Health Center 156

alerts

enabling 130

resolving

GET RECOMMENDATIONS command 149

Health Center 150

SQL queries 144

retrieving recommendations

client application 145

aliases

monitor elements

input\_db\_alias element 461

API request types

health monitor 184

snapshot monitor 97

applications

monitor elements

appl\_id 325

appl\_id\_holding\_lk 327

appl\_id\_oldest\_xact 327

appl\_idle\_time 328

appl\_name 328

appl\_priority 329

appl\_priority\_type 329

appl\_section\_inserts 330

appl\_section\_lookups 330

appl\_status 331

application\_handle 333

appls\_cur\_cons 334

appls\_in\_db2 334

client\_applname 354

creator 388

rolled\_back\_participant\_no 620

tpmon\_client\_app 726

attributes

monitor elements

progress\_list\_attr monitor element 600

audits

monitor elements

audit\_events\_total 336

audit\_file\_write\_wait\_time 336

audit\_file\_writes\_total 337

authorization ID

monitor elements

auth\_id 341

- authorization ID (*continued*)
  - monitor elements (*continued*)
    - execution\_id element 421
    - quiescer\_auth\_id 606
    - session\_auth\_id element 641
- authorization level
  - monitor elements
    - authority\_lvl 342
- automatic storage path
  - monitor elements
    - db\_storage\_path 395
    - sto\_path\_free\_sz 674

## B

- backup
  - health indicators
    - db.db\_backup\_req 175
  - monitor elements
    - last\_backup 474
  - requirements
    - health indicator 175
- books
  - printed
    - ordering 756
- buffer pools
  - monitor elements
    - activity 122
    - automatic 343
    - block\_ios 344
    - bp\_cur\_buffsz 347
    - bp\_id 347
    - bp\_name 347
    - bp\_new\_buffsz 347
    - bp\_pages\_left\_to\_remove 348
    - bp\_tbsp\_use\_count 348
    - buff\_free 348
    - buff\_free\_bottom 348
    - pool\_async\_data\_read\_reqs 546
    - pool\_async\_data\_reads 547
    - pool\_async\_data\_writes 548
    - pool\_async\_index\_read\_reqs 548
    - pool\_async\_index\_reads 549
    - pool\_async\_index\_writes 550
    - pool\_async\_read\_time 551
    - pool\_async\_write\_time 551
    - pool\_async\_xda\_read\_reqs 552
    - pool\_async\_xda\_reads 553
    - pool\_async\_xda\_writes 554
    - pool\_data\_l\_reads 556
    - pool\_data\_p\_reads 557
    - pool\_data\_writes 559
    - pool\_drty\_pg\_steal\_clns 561
    - pool\_drty\_pg\_thrsh\_clns 562
    - pool\_index\_l\_reads 564
    - pool\_index\_p\_reads 566
    - pool\_index\_writes 567
    - pool\_lsn\_gap\_clns 569
    - pool\_no\_victim\_buffer 570
    - pool\_read\_time 571
    - pool\_temp\_data\_l\_reads 573
    - pool\_temp\_data\_p\_reads 575
    - pool\_temp\_index\_l\_reads 577
    - pool\_temp\_index\_p\_reads 579
    - pool\_temp\_xda\_l\_reads 580
    - pool\_temp\_xda\_p\_reads 582
    - pool\_write\_time 584

- buffer pools (*continued*)
  - monitor elements (*continued*)
    - pool\_xda\_l\_reads 585
    - pool\_xda\_p\_reads 588
    - pool\_xda\_writes 589
    - tablespace\_cur\_pool\_id 682
    - tablespace\_next\_pool\_id 688
  - monitoring
    - administrative views 195
- BUFFERPOOLS event type
  - overview 9
- buffers
  - monitor elements
    - num\_log\_data\_found\_in\_buffer 523
- byte order
  - monitor elements
    - byte\_order 349

## C

- caching
  - stats\_cache\_size monitor element 658
- capturing health snapshots
  - using a client application 137
  - using CLP 136
  - using SQL 135
- catalog cache
  - health indicators
    - db.catcache\_hitratio 179
  - monitor elements
    - cat\_cache\_inserts 349
    - cat\_cache\_lookups 350
    - cat\_cache\_overflows 351
    - cat\_cache\_size\_top 351
- catalog nodes
  - monitor elements
    - catalog\_node 352
    - catalog\_node\_name 352
- CCSID (coded character set identifier)
  - monitor elements
    - host\_ccsid 457
- channels currently free monitor element
  - FCM (Fast Communications Manager)
    - ch\_free monitor element 353
- client applications
  - capturing health snapshots 137
- client operating platform monitor element
  - client\_platform element 357
- client process ID monitor element
  - client\_pid element 356
- client product and version ID monitor element
  - client\_prdid element 357
- CLP (command line processor)
  - capturing health snapshots 136
  - commands
    - health monitor 184
- code pages
  - monitor elements
    - codepage\_id 360
    - host\_ccsid 457
- commit
  - monitor elements
    - int\_commits element 463
- commit statements attempted monitor element
  - commit\_sql\_stmts element 361
- communication error time monitor element
  - gw\_comm\_error\_time element 442

- communication errors monitor element
  - gw\_comm\_errors element 442
- communication protocol
  - monitor elements
    - client\_protocol 358
- completed progress work units monitor element
  - monitor elements
    - progress\_completed\_units element 600
- con\_response\_time monitor element 363
- configuring
  - db2toprc 108
- connections
  - monitor elements
    - appl\_con\_time 324
    - appls\_cur\_cons 334
    - appls\_in\_db2 334
    - con\_elapsed\_time 363
    - con\_local\_databases 363
    - conn\_complete\_time 371
    - conn\_time 371
    - connection\_status 372
    - connections\_top 372
    - dl\_conns 414
    - gw\_connections\_top 443
    - gw\_cons\_wait\_client 443
    - gw\_cons\_wait\_host 444
    - gw\_cur\_cons 444
    - gw\_total\_cons 445
    - local\_cons 476
    - local\_cons\_in\_exec 477
    - num\_gw\_conn\_switches 521
    - rem\_cons\_in 610
    - rem\_cons\_in\_exec 610
    - total\_sec\_cons 717
- CONNECTIONS event type
  - overview 9
- containers
  - monitor elements
    - container\_accessible 373
    - container\_id 374
    - container\_name 374
    - container\_total\_pages 375
    - container\_type 375
    - container\_usable\_pages 376
- counters
  - data element type 118
- CPU time
  - monitor elements
    - agent\_sys\_cpu\_time 315
    - agent\_usr\_cpu\_time 316
    - ss\_sys\_cpu\_time 656
    - ss\_usr\_cpu\_time 657
    - stmt\_sys\_cpu\_time 669
    - stmt\_usr\_cpu\_time 671
    - system\_cpu\_time 677
    - total\_cpu\_time 712
    - total\_sys\_cpu\_time 723
    - total\_usr\_cpu\_time 725
    - user\_cpu\_time 739
- CREATE EVENT MONITOR statement
  - event types 9
- creator monitor element 388
- cursors
  - monitor elements
    - acc\_curs\_blk 302
    - blocking\_cursor 345
    - cursor\_name 389

- cursors (*continued*)
  - monitor elements (*continued*)
    - open\_cursors 527
    - open\_loc\_curs 528
    - open\_loc\_curs\_blk 528
    - open\_rem\_curs 528
    - open\_rem\_curs\_blk 529
    - rej\_curs\_blk 609
- custom controls
  - accessing 132

## D

- data
  - element types
    - counters 118
    - overview 117
  - data objects
    - monitoring 231
  - data partitions
    - monitor elements
      - data\_partition\_id 390
  - data sources
    - data source name monitor element 391
    - health indicator 182
  - DATABASE event type 9
  - database manager monitor elements
    - server\_db2\_type element 636
  - database monitor
    - overview 3
  - database paths
    - db\_path element monitor element 394
  - database system events
    - information collection 58
  - database system monitor
    - data organization 117
    - information
      - restricting 111
    - interfaces 124
    - memory requirements 119
    - output 119
    - sample 124
    - self-describing data stream 119
  - database-managed space (DMS)
    - table spaces
      - health indicators 164
  - databases
    - aliases
      - application monitor element 355
      - gateway monitor element 444
    - connections
      - connects since database activation monitor element 711
    - monitor elements
      - application 355
      - connects since database activation 711
      - database deactivation timestamp 413
      - gateway 444
  - datasource\_name element 391
  - db\_heap\_top monitor element
    - description 392
  - db.lock\_escal\_rate health indicator 178
  - db.locklist\_utilization health indicator 177
  - DB2 Connect
    - monitor elements
      - gw\_con\_time 443
      - gw\_cur\_cons 444



- DB2 Connect (*continued*)
  - monitor elements (*continued*)
    - gw\_exec\_time 445
    - gw\_total\_cons 445
- DB2 Information Center
  - languages 757
  - updating 758, 759
  - versions 757
  - viewing in different languages 757
- DB2 Performance Counters 208
- db2eventctl control file 70
- db2evmonfmt Java-based tool
  - description 16
- db2evmonfmt tool 23, 43
- db2perfc command
  - resetting database performance values 211
- db2perfi command
  - installing and registering DB2Perf.DLL 208
- db2perfr command
  - registering administrator username and password with DB2 209
- db2top
  - configuration file 108
- db2top monitoring utility 105
- deadlock reports 23
- deadlocks
  - db.deadlock\_rate health indicator 177
  - event types 9
  - monitor elements
    - deadlock\_id 398
    - deadlock\_node 399
    - deadlocks 399
    - dl\_conns 414
    - int\_deadlock\_rollbacks 464
    - participant\_no 540
- DELETE statement
  - delete\_sql\_stmts monitor element 401
- descriptors
  - progress\_description monitor element 600
- disconn\_time element 413
- documentation
  - overview 753
  - PDF 753
  - printed 753
  - terms and conditions of use 762

**E**

- environment handles
  - comp\_env\_desc monitor element 362
- errors
  - gw\_comm\_errors monitor element 442
- event monitors
  - blocked 71
  - buffers 71
  - creating
    - file 68
    - overview 60
    - partitioned database 73
    - pipe 71
    - table 61
  - data transferring between systems 78
  - database system events 58
  - db2evmonfmt Java-based tool for parsing data 16
  - DEADLOCK WITH DETAILS HISTORY 205
  - elements
    - count 384

- event monitors (*continued*)
  - elements (*continued*)
    - event\_monitor\_name 419
    - evmon\_activates 419
    - evmon\_flushes 420
  - event type to logical data group mappings 276
  - file management 70
  - named pipe management 72
  - non-blocked 71
  - output
    - sample 75
    - self-describing data stream 76
  - overview 58
  - records 76
  - table management 63
  - unformatted event table 16
- event records
  - finding corresponding applications 76
- events
  - monitor elements
    - event\_time 419
    - start\_time 657
    - stop\_time 674

## F

- Fast Communication Manager (FCM)
  - monitor elements
    - fcm\_message\_rcv\_volume 422
    - fcm\_message\_rcv\_wait\_time 423
- FCM (Fast Communications Manager)
  - monitor elements
    - buff\_free 348
    - buff\_free\_bottom 348
    - ch\_free 353
    - ch\_free\_bottom 353
    - total\_buffers\_rcvd 711
    - total\_buffers\_sent 711
- federated servers
  - monitor elements
    - disconnects element 414
- fetching
  - monitor elements
    - fetch\_count 438
- file event monitors
  - buffering 71
  - creating 68
  - file management 70
  - formatting output from command line 75
- file systems
  - health indicators
    - db.log\_fs\_util 176
  - monitor elements
    - fs\_caching 440
    - fs\_id 440
    - fs\_total\_size 441
    - fs\_type 441
    - fs\_used\_size 442
- files
  - monitor elements
    - files\_closed 438
- FLUSH EVENT MONITOR statement
  - event types 9
- format
  - health indicator 164

## G

- GET SNAPSHOT command
  - sample output 99, 196
- global health snapshots 141
- global snapshots on partitioned database systems 102
- graphical tools
  - health monitor 142
- gw\_db\_alias element 444

## H

- hash joins
  - monitor elements
    - active\_hash\_joins 308
    - hash\_join\_overflows 455
    - hash\_join\_small\_overflows 455
    - post\_shrthreshold\_hash\_joins 591
    - post\_threshold\_hash\_joins 593
    - total\_hash\_joins 713
- health alerts
  - enabling 130
  - recommendations
    - retrieving using CLP 145
  - resolving
    - client applications 149
    - SQL queries 144
- Health Center
  - health indicators 127, 159
  - interface 132
  - overview 132, 142
  - status beacon 142
  - tasks 132
- health indicators
  - alert actions
    - combined states 158
  - alerts
    - resolving using SQL 144
    - resolving using the Health Center 150
    - retrieving recommendations 145, 149
  - applications waiting on locks 179
  - catalog cache hit ratio 179
  - collection state-based 127, 159
  - configuration
    - client applications 154
    - Health Center 156
    - overview 150
    - resetting 154
    - retrieving 152
    - updates 153
  - data 135
  - databases
    - heap utilization 181
    - highest severity alert state 173
    - operational state 173
  - db.alert\_state 173
  - db.apps\_waiting\_locks 179
  - db.catcache\_hitratio 179
  - db.database\_heap\_util 181
  - db.db\_auto\_storage\_util 165
  - db.db\_backup\_req 175
  - db.db\_op\_status 173
  - db.deadlock\_rate 177
  - db.fed\_nicknames\_op\_status 181
  - db.fed\_servers\_op\_status 182
  - db.hadr\_delay 175
  - db.hadr\_op\_status 175
  - health indicators (*continued*)
    - db.lock\_escal\_rate 178
    - db.locklist\_utilization 177
    - db.log\_fs\_util 176
    - db.log\_util 176
    - db.max\_sort\_shrmem\_util 171
    - db.pkgcache\_hitratio 180
    - db.shrworkspace\_hitratio 180
    - db.sort\_shrmem\_util 170
    - db.spilled\_sorts 171
    - db.tb\_reorg\_req 174
    - db.tb\_runstats\_req 174
    - db2.db2\_alert\_state 172
    - db2.db2\_op\_status 172
    - db2.mon\_heap\_util 180
    - db2.sort\_privmem\_util 169
    - deadlock rate 177
    - DMS table spaces 164
    - format 164
    - instances
      - highest severity alert state 172
      - operational state 172
    - lock escalation rate 178
    - lock list utilization 177
    - logs
      - file system utilization 176
      - space utilization 176
    - monitor heap utilization 180
    - overview 127, 159
    - package cache hit ratio 180
    - process cycle 129
    - shared workspace hit ratio 180
    - sort memory utilization
      - long-term shared 171
      - private 169
      - shared 170
    - sorts that overflowed 171
    - state-based 127, 159
    - summary 161
    - table spaces
      - container operational state 169
      - container utilization 168
      - operational state 169
      - storage utilization 167
    - threshold-based 127, 159
    - ts.ts\_auto\_resize\_status 166
    - ts.ts\_op\_status 169
    - ts.ts\_util 167
    - ts.ts\_util\_auto\_resize 166
    - tsc.tscont\_op\_status 169
    - tsc.utilization 168
- health monitor
  - alerts 150
  - API request types 184
  - CLP commands 184
  - description 127
  - graphical tools 142
  - Health Center 142
  - Health Center Status Beacon 142
  - interfaces 182
  - logical data groups 160
  - recommendation retrieval
    - using client application 149
    - using CLP 145
    - using SQL 144
  - sample output 140
  - SQL table functions 183

- health monitor *(continued)*
  - starting 134
  - stopping 134
  - thresholds 150
- health snapshots
  - capturing
    - using client applications 137
    - using CLP 136
    - using SQL table functions 135
  - global 141
- help
  - configuring language 757
  - SQL statements 757
- High Availability Disaster Recovery (HADR)
  - health indicators
    - db.hadr\_delay 175
    - db.hadr\_op\_status 175
  - monitor elements
    - hadr\_connect\_status 445
    - hadr\_connect\_time 446
    - hadr\_heartbeat 447
    - hadr\_local\_host 447
    - hadr\_local\_service 448
    - hadr\_log\_gap 448
    - hadr\_peer\_window 449
    - hadr\_peer\_window\_end 449
    - hadr\_primary\_log\_file 450
    - hadr\_primary\_log\_lsn 450
    - hadr\_primary\_log\_page 450
    - hadr\_remote\_host 451
    - hadr\_remote\_instance 451
    - hadr\_remote\_service 451
    - hadr\_role 452
    - hadr\_standby\_log\_file 452
    - hadr\_standby\_log\_lsn 453
    - hadr\_standby\_log\_page 453
    - hadr\_state 453
    - hadr\_syncmode 454
    - hadr\_timeout 454
- histograms
  - monitor elements
    - histogram\_type 456
    - number\_in\_bin 526
    - top 708
- host databases
  - host\_db\_name monitor element 457
  - name monitor element 457

## I

- I/O
  - monitor elements
    - num\_log\_part\_page\_io 523
    - num\_log\_read\_io 524
    - num\_log\_write\_io 524
    - num\_pages\_from\_block\_IOs element 537
    - num\_pages\_from\_vectored\_IOs element 537
    - vectored\_ios 742
- identifiers
  - monitor elements
    - arm\_correlator 335
    - bin\_id 343
    - db\_work\_action\_set\_id 396
    - db\_work\_class\_id 396
    - host\_prdid element 457
    - sc\_work\_action\_set\_id 631
    - sc\_work\_class\_id 631

- identifiers *(continued)*
  - monitor elements *(continued)*
    - service\_class\_id 639
    - sql\_req\_id element 649
    - work\_action\_set\_id 744
    - work\_class\_id 745
- indexes
  - index object pages monitor element 460
  - monitor elements
    - iid 459, 621
    - index\_object\_pages element 460
    - index\_only\_scans 460
    - index\_scans 460
    - index\_tbsp\_id 461
    - int\_node\_splits 465
    - nleaf 517
    - nlevels 517
    - page\_allocations 536
    - pages\_merged 538
    - reorg\_index\_id monitor element 612
- Indoubt Transaction Manager
  - overview 211
- indoubt transactions
  - monitoring 211
- insert\_timestamp monitor element 462
- inserting data
  - monitor elements
    - appl\_section\_inserts 330
- instances
  - operational states
    - health indicator 172
- int\_rows\_deleted monitor element 466
- isolation levels
  - monitor elements
    - effective\_isolation 415

## J

- java tools
  - db2evmonfmt 23, 43

## L

- LOBs (large objects)
  - lob\_object\_pages element 476
- local databases
  - monitor elements
    - con\_local\_databases 363
- location
  - monitor elements
    - db\_location element 393
- lock list utilization health indicator 177
- lock modes
  - monitor elements
    - lock\_current\_mode monitor element 479
    - lock\_mode element 483
    - lock\_mode\_requested element 484
- lock timeout reports 23
- lock wait reports 23
- lock waits
  - monitor elements
    - lock\_wait\_start\_time 490
- lock\_escalation element 480
- locking event monitor
  - written to relational table 38

- locks
  - escalation
    - escalation monitor element 480
    - health indicator 178
  - monitor elements
    - agent\_id\_holding\_lock 314
    - appl\_id\_holding\_lk 327
    - effective\_lock\_timeout 416
    - lock\_attributes 478
    - lock\_count 478
    - lock\_escals 480
    - lock\_hold\_count 482
    - lock\_list\_in\_use 482
    - lock\_name monitor element 484
    - lock\_node element 485
    - lock\_object\_name element 485
    - lock\_object\_type element 486
    - lock\_release\_flags monitor element 487
    - lock\_status element 487
    - lock\_timeout\_val element 488
    - lock\_timeouts 489
    - lock\_wait\_time 490
    - lock\_waits 492
    - locks\_held 494
    - locks\_held\_top 494
    - locks\_in\_list 495
    - locks\_waiting 495
    - participant\_no\_holding\_lk 540
    - remote\_lock\_time 611
    - remote\_locks 611
    - sequence\_no\_holding\_lk 636
    - stmt\_lock\_timeout 663
    - uow\_lock\_wait\_time 735
    - x\_lock\_escals 748
- log buffer
  - monitor elements
    - num\_log\_buffer\_full 522
- log disk
  - monitor elements
    - log\_disk\_wait\_time 496
    - log\_disk\_waits\_total 497
- log files
  - health indicators
    - db.log\_fs\_util 176
  - monitor elements
    - current\_active\_log 388
    - current\_archive\_log 389
    - diaglog\_write\_wait\_time 403
    - diaglog\_writes\_total 403
    - first\_active\_log 439
    - hadr\_log\_gap 448
    - hadr\_primary\_log\_file 450
    - hadr\_primary\_log\_page 450
    - hadr\_standby\_log\_file 452
    - hadr\_standby\_log\_page 453
    - last\_active\_log 473
    - log\_read\_time 499
    - log\_reads 499
    - sec\_logs\_allocated element 632
- log sequence number (LSN)
  - monitor elements
    - hadr\_primary\_log\_lsn 450
    - hadr\_standby\_log\_lsn 453
- log spaces
  - health indicators
    - db.log\_util 176

- log spaces (*continued*)
  - monitor elements
    - log\_held\_by\_dirty\_pages 498
    - log\_to\_redo\_for\_recovery 500
    - log\_write\_time 500
    - log\_writes 501
    - sec\_log\_used\_top element 631
    - smallest\_log\_avail\_node 644
    - tot\_log\_used\_top element 709
    - total\_log\_available element 714
    - total\_log\_used element 715
    - uow\_log\_space\_used 735
  - logical data groups
    - COLLECT ACTIVITY DATA settings effects 300
    - data organization 117
    - event monitors 279
    - health monitor 160
    - mapping to event types 276
    - snapshot monitor 241
  - long data
    - monitor elements
      - long\_object\_pages element 501

## M

- memory
  - health indicators
    - db.sort\_shrmem\_util 170
    - db2.sort\_privmem\_util 169
  - monitor elements
    - comm\_private\_mem 361
    - db\_heap\_top 392
    - lock\_list\_in\_use 482
    - pool\_cur\_size 555
    - pool\_id 563
    - pool\_max\_size 555
    - pool\_secondary\_id 573
    - pool\_watermark 583
  - memory requirements
    - database system monitor 119
  - Memory Visualizer
    - overview 187
    - using 185
  - messages
    - monitor elements
      - message 515
      - message\_time 516
  - metrics
    - data objects 231
  - minimum channels free monitor element
    - FCM (Fast Communications Manager)
      - ch\_free\_bottom monitor element 353
  - mon\_heap\_sz database manager configuration parameter 119
  - monitor element
    - num\_exec\_with\_metrics 520
    - table reorganization
      - reorg\_xml\_regions\_compressed 617
      - reorg\_xml\_regions\_rejected\_for\_compression 617
  - monitor elements
    - acc\_curs\_blk 302
    - act\_remapped\_in 306
    - act\_remapped\_out 306
    - activation time
      - last\_wlm\_reset 475
    - active\_sorts 308
    - activities
      - act\_aborted\_total 302

- monitor elements (*continued*)
  - activities (*continued*)
    - act\_completed\_total 303
    - act\_rejected\_total 305
    - act\_total 307
    - activity\_collected 309
    - activity\_id 309
    - activity\_secondary\_id 310
    - activity\_state 310
    - activity\_type 311
    - coord\_act\_aborted\_total 376
    - coord\_act\_completed\_total 376
    - coord\_act\_rejected\_total 382
    - parent\_activity\_id 538
  - activity 229
  - activity\_metrics 54
  - ACTIVITYTOTALTIME activity threshold
    - activitytotaltime\_threshold\_id 311
    - activitytotaltime\_threshold\_value 312
    - activitytotaltime\_threshold\_violated 312
  - address 312
  - agents
    - agent\_id 313
    - agent\_id\_holding\_lock 314
    - agent\_pid 315
    - agent\_status 315
    - agent\_sys\_cpu\_time 315
    - agent\_usr\_cpu\_time 316
    - agent\_wait\_time 317
    - agent\_waits\_total 318
    - agents\_created\_empty\_pool 319
    - agents\_from\_pool 319
    - agents\_registered 320
    - agents\_registered\_top 320
    - agents\_stolen 320
    - agents\_top 321
    - agents\_waiting\_on\_token 321
    - agents\_waiting\_top 321
    - appl\_priority 329
    - associated\_agents\_top 335
    - coord\_agent\_pid 382
    - coord\_agents\_top 383
    - idle\_agents 458
    - max\_agent\_overflows 502
    - num\_agents 518
    - num\_assoc\_agents 519
    - priv\_workspace\_size\_top element 599
    - quiescer\_agent\_id 605
    - rolled\_back\_agent\_id 620
  - agg\_temp\_tablespace\_top 322
  - aliases
    - client\_db\_alias 355
    - input\_db\_alias element 461
  - applications
    - appl\_id 325
    - appl\_id\_holding\_lk 327
    - appl\_id\_oldest\_xact 327
    - appl\_idle\_time 328
    - appl\_name 328
    - appl\_priority\_type 329
    - appl\_section\_inserts 330
    - appl\_section\_lookups 330
    - appl\_status 331
    - application\_handle 333
    - client\_applname 354
    - tpmon\_client\_app 726

- monitor elements (*continued*)
  - attributes
    - progress\_list\_attr monitor element 600
  - audits
    - audit\_events\_total 336
    - audit\_file\_write\_wait\_time 336
    - audit\_file\_writes\_total 337
  - auth\_id 341
  - authority\_bitmap 341
  - authorization ID
    - execution\_id element 421
    - session\_auth\_id element 641
  - automatic storage path
    - sto\_path\_free\_sz 674
  - binds\_precompiles 344
  - blocking\_cursor 345
  - blocks\_pending\_cleanup 346
  - boundary\_leaf\_node\_splits 346
  - buffer pools
    - activity 122
    - automatic 343
    - block\_ios 344
    - bp\_cur\_buffsz 347
    - bp\_id 347
    - bp\_name 347
    - bp\_new\_buffsz 347
    - bp\_pages\_left\_to\_remove 348
    - bp\_tbsp\_use\_count 348
    - buff\_free 348
    - buff\_free\_bottom 348
    - pool\_async\_data\_read\_reqs 546
    - pool\_async\_data\_reads 547
    - pool\_async\_data\_writes 548
    - pool\_async\_index\_read\_reqs 548
    - pool\_async\_index\_reads 549
    - pool\_async\_index\_writes 550
    - pool\_async\_read\_time 551
    - pool\_async\_write\_time 551
    - pool\_async\_xda\_read\_reqs 552
    - pool\_async\_xda\_reads 553
    - pool\_async\_xda\_writes 554
    - pool\_data\_l\_reads 556
    - pool\_data\_p\_reads 557
    - pool\_data\_writes 559
    - pool\_drty\_pg\_steal\_clns 561
    - pool\_drty\_pg\_thrsh\_clns 562
    - pool\_index\_l\_reads 564
    - pool\_index\_p\_reads 566
    - pool\_index\_writes 567
    - pool\_lsn\_gap\_clns 569
    - pool\_no\_victim\_buffer 570
    - pool\_read\_time 571
    - pool\_temp\_data\_l\_reads 573
    - pool\_temp\_data\_p\_reads 575
    - pool\_temp\_index\_l\_reads 577
    - pool\_temp\_index\_p\_reads 579
    - pool\_temp\_xda\_l\_reads 580
    - pool\_temp\_xda\_p\_reads 582
    - pool\_write\_time 584
    - pool\_xda\_l\_reads 585
    - pool\_xda\_p\_reads 588
    - pool\_xda\_writes 589
  - buffers
    - num\_log\_data\_found\_in\_buffer 523
  - byte order
    - byte\_order 349

monitor elements (*continued*)

- caching
  - stats\_cache\_size 658
- cat\_cache\_inserts 349
- cat\_cache\_lookups 350
- cat\_cache\_overflows 351
- cat\_cache\_size\_top 351
- catalog\_node 352
- catalog\_node\_name 352
- client\_pid element 356
- client\_platform element 357
- client\_prdid element 357
- code pages
  - codepage\_id 360
  - host\_ccsid 457
- comm\_private\_mem 361
- commit
  - int\_commits element 463
- commit\_sql\_stmts element 361
- communication protocol
  - client\_protocol 358
- completed progress work units monitor element
  - progress\_completed\_units element 600
- connections
  - appl\_con\_time 324
  - appls\_cur\_cons 334
  - appls\_in\_db2 334
  - con\_elapsed\_time 363
  - con\_local\_databases 363
  - conn\_complete\_time 371
  - conn\_time 371
  - connection\_status 372
  - connections\_top 372
  - gw\_connections\_top 443
  - gw\_cons\_wait\_client 443
  - gw\_cons\_wait\_host 444
  - gw\_cur\_cons 444
  - gw\_total\_cons 445
  - local\_cons 476
  - local\_cons\_in\_exec 477
  - num\_gw\_conn\_switches 521
  - rem\_cons\_in 610
  - rem\_cons\_in\_exec 610
  - total\_sec\_cons 717
- containers
  - container\_accessible 373
  - container\_id 374
  - container\_name 374
  - container\_total\_pages 375
  - container\_type 375
  - container\_usable\_pages 376
- coord\_act\_est\_cost\_avg 377
- coord\_act\_exec\_time\_avg 378
- coord\_act\_interarrival\_time\_avg 378
- coord\_act\_lifetime\_avg 379
- coord\_act\_queue\_time\_avg 381
- coord\_member 381
- country\_code
  - replaced with territory\_code 704
- CPU time
  - ss\_sys\_cpu\_time 656
  - ss\_usr\_cpu\_time 657
  - stmt\_sys\_cpu\_time 669
  - stmt\_usr\_cpu\_time 671
  - system\_cpu\_time 677
  - total\_cpu\_time 712
  - total\_sys\_cpu\_time 723

monitor elements (*continued*)

- CPU time (*continued*)
  - total\_usr\_cpu\_time 725
  - user\_cpu\_time 739
- cursors
  - cursor\_name 389
  - rej\_curs\_blk 609
- data organization 117
- database connections
  - total\_cons element 711
- database manager
  - server\_db2\_type element 636
- database paths
  - db\_path element 394
- database system 301
- db\_heap\_top 392
- db\_storage\_path 395
- DB2 Connect
  - gw\_con\_time 443
  - gw\_exec\_time 445
- deadlocks
  - deadlock\_id 398
  - deadlock\_node 399
  - deadlocks 399
  - dl\_conns 414
  - int\_deadlock\_rollbacks 464
- del\_keys\_cleaned 401
- DELETE statement
  - delete\_sql\_stmts element 401
- deleting
  - int\_rows\_deleted element 466
- descriptors
  - progress\_description element 600
- destination\_service\_class\_id 402
- eff\_stmt\_text 415
- empty\_pages\_deleted 417
- empty\_pages\_reused 417
- environment handles
  - comp\_env\_desc element 362
- errors
  - gw\_comm\_errors element 442
- event monitors
  - count 384
  - event\_monitor\_name 419
  - evmon\_activates 419
  - evmon\_flushes 420
  - list 279
- events
  - event\_time element 419
  - start\_time element 657
  - stop\_time element 674
- executable\_id 420
- executing
  - act\_exec\_time 305
- fabrications
  - stats\_fabricate\_time 659
  - stats\_fabrications 660
- Fast Communication Manager (FCM)
  - fcm\_message\_rcv\_volume 422
  - fcm\_message\_rcv\_wait\_time 423
- FCM (Fast Communications Manager)
  - ch\_free monitor element 353
  - ch\_free\_bottom monitor element 353
  - total\_buffers\_rcvd element 711
  - total\_buffers\_sent element 711
- federated servers
  - disconnects element 414

monitor elements (*continued*)

- fetching
  - fetch\_count 438
- file systems
  - fs\_caching 440
  - fs\_id 440
  - fs\_total\_size 441
  - fs\_type 441
  - fs\_used\_size 442
- files
  - files\_closed 438
- gw\_comm\_error\_time element 442
- HADR (high availability disaster recovery)
  - hadr\_connect\_status 445
  - hadr\_connect\_time 446
  - hadr\_heartbeat 447
  - hadr\_local\_host 447
  - hadr\_local\_service 448
  - hadr\_log\_gap 448
  - hadr\_peer\_window 449
  - hadr\_peer\_window\_end 449
  - hadr\_primary\_log\_file 450
  - hadr\_primary\_log\_lsn 450
  - hadr\_primary\_log\_page 450
  - hadr\_remote\_host 451
  - hadr\_remote\_instance 451
  - hadr\_remote\_service 451
  - hadr\_role 452
  - hadr\_standby\_log\_file 452
  - hadr\_standby\_log\_lsn 453
  - hadr\_standby\_log\_page 453
  - hadr\_state 453
  - hadr\_syncmode 454
  - hadr\_timeout 454
- hash joins
  - active\_hash\_joins 308
  - hash\_join\_overflows 455
  - hash\_join\_small\_overflows 455
  - post\_shrthreshold\_hash\_joins 591
  - post\_threshold\_hash\_joins 593
  - total\_hash\_joins 713
- histograms
  - histogram\_type 456
  - number\_in\_bin 526
  - top 708
- host databases
  - host\_db\_name element 457
- I/O
  - num\_log\_part\_page\_io 523
  - num\_log\_read\_io 524
  - num\_log\_write\_io 524
  - num\_pages\_from\_block\_IOs element 537
  - num\_pages\_from\_vectored\_IOs element 537
  - vectored\_ios 742
- identifiers
  - arm\_correlator 335
  - bin\_id 343
  - db\_work\_action\_set\_id 396
  - db\_work\_class\_id 396
  - host\_prdid element 457
  - sc\_work\_action\_set\_id 631
  - sc\_work\_class\_id 631
  - service\_class\_id 639
  - sql\_req\_id element 649
  - work\_action\_set\_id 744
  - work\_class\_id 745
- inbound\_bytes\_received element 459

monitor elements (*continued*)

- inbound\_bytes\_sent element 459
- inbound\_comm\_address element 459
- include\_col\_updates 460
- indexes
  - iid 459, 621
  - index\_object\_pages element 460
  - index\_only\_scans 460
  - index\_scans 460
  - index\_fbsp\_id 461
  - int\_node\_splits 465
  - nleaf 517
  - nlevels 517
  - page\_allocations 536
  - pages\_merged 538
- insert\_timestamp 462
- is\_system\_appl 473
- isolation levels
  - effective\_isolation 415
- key\_updates 473
- LOBs (large objects)
  - lob\_object\_pages element 476
- location
  - db\_location element 393
- lock modes
  - lock\_current\_mode monitor element 479
  - lock\_mode element 483
  - lock\_mode\_requested element 484
- locking 235
- locks
  - effective\_lock\_timeout 416
  - lock\_attributes 478
  - lock\_count 478
  - lock\_escals 480
  - lock\_hold\_count 482
  - lock\_list\_in\_use 482
  - lock\_name monitor element 484
  - lock\_node element 485
  - lock\_object\_name element 485
  - lock\_object\_type element 486
  - lock\_release\_flags monitor element 487
  - lock\_status element 487
  - lock\_timeout\_val element 488
  - lock\_timeouts 489
  - lock\_wait\_time 490
  - lock\_waits 492
  - locks\_held 494
  - locks\_held\_top 494
  - locks\_in\_list 495
  - locks\_waiting 495
  - participant\_no\_holding\_lk 540
  - remote\_lock\_time 611
  - remote\_locks 611
  - sequence\_no\_holding\_lk 636
  - stmt\_lock\_timeout 663
  - uow\_lock\_wait\_time 735
  - x\_lock\_escals 748
- log buffers
  - num\_log\_buffer\_full 522
- log disk
  - log\_disk\_wait\_time 496
  - log\_disk\_waits\_total 497
- log files
  - current\_active\_log 388
  - current\_archive\_log 389
  - diaglog\_write\_wait\_time 403
  - diaglog\_writes\_total 403



monitor elements (*continued*)

- log files (*continued*)
  - first\_active\_log 439
  - last\_active\_log 473
  - log\_read\_time 499
  - log\_reads 499
  - sec\_logs\_allocated element 632
- log spaces
  - log\_held\_by\_dirty\_pages 498
  - log\_to\_redo\_for\_recovery 500
  - log\_write\_time 500
  - log\_writes 501
  - sec\_log\_used\_top element 631
  - smallest\_log\_avail\_node 644
  - tot\_log\_used\_top element 709
  - total\_log\_available element 714
  - total\_log\_used element 715
  - uow\_log\_space\_used 735
- logical data groups 245
- long data
  - long\_object\_pages element 501
- member 514
- messages
  - message 515
- names
  - db\_name element 393
  - dcs\_db\_name element 397
  - service\_subclass\_name 640
  - service\_superclass\_name 640
  - work\_action\_set\_name 745
  - work\_class\_name 745
- network time
  - max\_network\_time\_1\_ms 513
  - max\_network\_time\_100\_ms 512
  - max\_network\_time\_16\_ms 512
  - max\_network\_time\_4\_ms 513
  - max\_network\_time\_500\_ms 513
  - max\_network\_time\_gt500\_ms 514
- network\_time\_bottom 516
- network\_time\_top 517
- nicknames
  - create\_nickname element 387
  - create\_nickname\_time element 387
- nodes
  - coord\_node monitor element 383
  - node\_number element 518
  - num\_nodes\_in\_db2\_instance 524
  - ss\_node\_number element 655
- nonboundary\_leaf\_node\_splits 518
- num\_db\_storage\_paths 519
- num\_indoubt\_trans 521
- num\_nodes\_in\_db2\_instance 524
- num\_remaps 525
- num\_transmissions 525
- num\_transmissions\_group 526
- numbers
  - progress\_list\_cur\_seq\_num element 601
  - ss\_number element 655
- OLAP functions 308, 527, 593, 715
- open\_cursors 527
- open\_loc\_curs 528
- open\_loc\_curs\_blk 528
- open\_rem\_curs 528
- open\_rem\_curs\_blk 529
- operations
  - direct\_read\_reqs element 404
  - direct\_read\_time element 406

monitor elements (*continued*)

- operations (*continued*)
  - direct\_reads element 407
  - direct\_write\_reqs element 409
  - direct\_write\_time element 410
  - direct\_writes element 412
  - stmt\_operation element 664
- outbound bytes
  - max\_data\_sent\_1024 507
  - max\_data\_sent\_128 507
  - max\_data\_sent\_16384 508
  - max\_data\_sent\_2048 508
  - max\_data\_sent\_256 508
  - max\_data\_sent\_31999 509
  - max\_data\_sent\_4096 509
  - max\_data\_sent\_512 510
  - max\_data\_sent\_64000 510
  - max\_data\_sent\_8192 511
  - max\_data\_sent\_gt64000 511
- outbound bytes received
  - max\_data\_received\_1024 502
  - max\_data\_received\_128 503
  - max\_data\_received\_16384 503
  - max\_data\_received\_2048 503
  - max\_data\_received\_256 504
  - max\_data\_received\_31999 504
  - max\_data\_received\_4096 505
  - max\_data\_received\_512 505
  - max\_data\_received\_64000 506
  - max\_data\_received\_8192 506
  - max\_data\_received\_gt64000 506
  - outbound\_bytes\_received 530
  - outbound\_bytes\_received\_bottom 531
  - outbound\_bytes\_received\_top 531
- outbound bytes sent
  - outbound\_bytes\_sent 531
  - outbound\_bytes\_sent\_bottom 532
  - outbound\_bytes\_sent\_top 532
- outbound communication
  - outbound\_appl\_id 529
  - outbound\_comm\_address 532
  - outbound\_comm\_protocol 533
- outbound sequence
  - outbound\_sequence\_no 533
- overflow records
  - first\_overflow\_time element 440
  - last\_over\_flow\_time element 474
  - overflow\_accesses 533
  - overflow\_creates 534
- package cache
  - pkg\_cache\_inserts 543
  - pkg\_cache\_lookups 543
  - pkg\_cache\_num\_overflow 545
  - pkg\_cache\_size\_top 545
- packages
  - package\_name 534
  - package\_schema 535
  - package\_version\_id 535
- pages
  - data\_object\_pages element 390
- parallelism
  - degree\_parallelism 401
- participant\_no 540
- partition information
  - partition\_number monitor element 541
- partitions
  - coord\_partition\_num 383



monitor elements (*continued*)

- partitions (*continued*)
  - data\_partition\_id 390
- pass-through
  - passthru 541
  - passthru\_time 541
- pool\_cur\_size 555
- pool\_id 563
- pool\_max\_size 555
- pool\_secondary\_id 573
- pool\_watermark 583
- prefetching
  - unread\_prefetch\_pages 733
- priv\_workspace\_num\_overflows element 597
- progress\_work\_metric element 602
- pseudo\_deletes 602
- pseudo\_empty\_pages 603
- queries
  - query\_card\_estimate 603
  - query\_cost\_estimate 604
  - queue\_assignments\_total 604
  - queue\_size\_top 605
  - queue\_time\_total 605
  - select\_time 635
- quiescer
  - quiescer\_auth\_id 606
  - quiescer\_obj\_id 606
  - quiescer\_state 606
  - quiescer\_ts\_id 606
- ranges
  - bottom 346
  - range\_adjustment element 607
  - range\_container\_id element 607
  - range\_end\_stripe element 607
  - range\_max\_extent element 607
  - range\_max\_page\_number element 608
  - range\_num\_containers 608
  - range\_number element 608
  - range\_offset element 608
  - range\_start\_stripe element 608
  - range\_stripe\_set\_number element 609
- rebalancing
  - current\_extent 389
- rebinding
  - int\_auto\_rebinds element 463
- records
  - partial\_record element 539
- reoptimization
  - stmt\_value\_isreopt 673
- reorg\_completion element 611
- reorg\_long\_tbsp\_id 613
- reorg\_tbsp\_id 616
- reorganization
  - page\_reorgs element 536
  - reorg\_current\_counter element 612
  - reorg\_max\_phase element 613
  - reorg\_phase monitor element 613
  - reorg\_phase\_start element 614
  - reorg\_rows\_compressed monitor element 614
  - reorg\_rows\_rejected\_for\_compression monitor element 615
  - reorg\_start element 615
  - reorg\_status element 615
  - reorg\_type element 616
- request\_exec\_time\_avg 617
- requests
  - rqsts\_completed\_total 630

monitor elements (*continued*)

- response time
  - delete\_time element 402
  - host\_response\_time element 458
  - insert\_time element 462
- roll-forward recovery
  - rf\_log\_num 617
  - rf\_status 618
  - rf\_timestamp 618
  - rf\_type 618
- rollback
  - int\_rollbacks 465
  - rollback\_sql\_stmts 619
  - rolled\_back\_appl\_id 620
  - rolled\_back\_participant\_no 620
  - rolled\_back\_sequence\_no 621
- routines
  - routine\_id 621
- rows
  - int\_rows\_inserted element 467
  - int\_rows\_updated element 467
  - rows\_deleted 621
  - rows\_fetched 622
  - rows\_inserted 623
  - rows\_modified 623
  - rows\_read 624
  - rows\_returned 626
  - rows\_selected element 628
  - rows\_updated 628
  - rows\_written element 629
  - sp\_rows\_selected element 648
- RUNSTATS utility
  - async\_runstats 335
  - sync\_runstats 675
  - sync\_runstats\_time 676
- section\_type 634
- sections
  - priv\_workspace\_section\_inserts element 597
  - priv\_workspace\_section\_lookups element 598
  - section\_env 632
  - section\_number element 633
- sequences
  - progress\_seq\_num element 601
  - sequence\_no 635
- servers
  - product\_name 599
  - server\_instance\_name 637
  - server\_platform 637
  - server\_prdid 637
  - server\_version 638
- service levels
  - service\_level 639
- service subclass
  - total\_rqst\_mapped\_in 716
  - total\_rqst\_mapped\_out 716
- shared workspace
  - shr\_workspace\_num\_overflows 642
  - shr\_workspace\_section\_inserts 642
  - shr\_workspace\_section\_lookups 643
  - shr\_workspace\_size\_top 643
- snapshots
  - time\_stamp element 708
- sorting
  - pipedsorts\_accepted element 542
  - pipedsorts\_requested element 542
  - post\_shrthreshold\_sorts monitor element 592
  - post\_threshold\_sorts element 594

monitor elements (*continued*)

  sorting (*continued*)

    sort\_heap\_allocated element 644  
    sort\_heap\_top monitor element 645  
    sort\_overflows element 645  
    sort\_shrheap\_allocated monitor element 647  
    sort\_shrheap\_top monitor element 647  
    total\_section\_sort\_proc\_time 718  
    total\_section\_sort\_time 719  
    total\_section\_sorts 717  
    total\_sorts element 721, 723

  source\_service\_class\_id 648

  SQL communications area (SQLCA)

    sqlca 651

  SQL operations

    elapsed\_exec\_time element 416

  SQL statements

    ddl\_sql\_stmts element 397  
    dynamic\_sql\_stmts element 414  
    failed\_sql\_stmts element 421  
    insert\_sql\_stmts element 461  
    num\_compilation element 519  
    num\_executions element 520  
    select\_sql\_stmts element 634  
    sql\_chains 649  
    sql\_reqs\_since\_commit element 650  
    sql\_stmts 650  
    static\_sql\_stmts element 657  
    stmt\_pkgcache\_id element 666  
    stmt\_query\_id element 666  
    stmt\_sorts element 667  
    stmt\_source\_id element 668  
    stmt\_text element 669  
    stmt\_value\_data element 672  
    stmt\_value\_index element 672  
    stmt\_value\_isnull element 673  
    stmt\_value\_type element 673  
    total\_exec\_time element 713  
    uid\_sql\_stmts element 732

  statements

    prep\_time\_best element 596  
    prep\_time\_worst element 596  
    stmt\_first\_use\_time element 661  
    stmt\_history\_id element 661  
    stmt\_history\_list\_size element 662  
    stmt\_invocation\_id element 662  
    stmt\_isolation element 662  
    stmt\_last\_use\_time 663  
    stmt\_nest\_level element 664  
    stmt\_node\_number element 664  
    stmt\_type element 670

  status

    db\_status element 394  
    db2\_status element 391  
    dcs\_appl\_status element 397  
    ss\_status element 656

  storage paths

    num\_db\_storage\_paths 519

  stored procedures

    stored\_proc\_time element 675  
    stored\_procs element 675

  stripe sets

    container\_stripe\_set 374

  system\_metrics 49

  table queues

    tq\_tot\_send\_spills 731

monitor elements (*continued*)

  table reorganization

    reorg\_end element 612

  table spaces

    index\_tbsp\_id 461  
    long\_tbsp\_id 501  
    tablespace\_auto\_resize\_enabled 681  
    tablespace\_content\_type 682  
    tablespace\_cur\_pool\_id 682  
    tablespace\_current\_size 683  
    tablespace\_extent\_size 683  
    tablespace\_free\_pages 683  
    tablespace\_id 684  
    tablespace\_increase\_size 684  
    tablespace\_increase\_size\_percent 685  
    tablespace\_initial\_size 685  
    tablespace\_last\_resize\_failed 685  
    tablespace\_last\_resize\_time 686  
    tablespace\_max\_size 686  
    tablespace\_min\_recovery\_time 686  
    tablespace\_name 687  
    tablespace\_next\_pool\_id 688  
    tablespace\_num\_containers 688  
    tablespace\_num\_quiescers 688  
    tablespace\_num\_ranges 689  
    tablespace\_page\_size 689  
    tablespace\_page\_top 689  
    tablespace\_pending\_free\_pages 690  
    tablespace\_prefetch\_size 690  
    tablespace\_rebalancer\_extents\_processed 691  
    tablespace\_rebalancer\_extents\_remaining 691  
    tablespace\_rebalancer\_last\_extent\_moved 692  
    tablespace\_rebalancer\_mode 692  
    tablespace\_rebalancer\_priority 693  
    tablespace\_rebalancer\_restart\_time 693  
    tablespace\_rebalancer\_start\_time 694  
    tablespace\_state 694  
    tablespace\_state\_change\_object\_id 695  
    tablespace\_state\_change\_ts\_id 695  
    tablespace\_total\_pages 696  
    tablespace\_type 696  
    tablespace\_usable\_pages 697  
    tablespace\_used\_pages 697  
    tablespace\_using\_auto\_storage 698  
    tbsp\_max\_page\_top 698  
    ts\_name 732

  tables

    tab\_file\_id 677  
    tab\_type 677  
    table\_file\_id element 678  
    table\_name element 678  
    table\_scans 679  
    table\_schema element 679  
    table\_type element 681

  TCP/IP

    tcpcip\_sends\_total 703

  territory\_code 704

  thresholds

    num\_threshold\_violations 525  
    threshold\_action 704  
    threshold\_domain 705  
    threshold\_maxvalue 705  
    threshold\_name 705  
    threshold\_predicate 706  
    threshold\_queuesize 706  
    thresholdid 706

- monitor elements (*continued*)
  - time
    - prefetch\_wait\_time element 595
    - prep\_time 595
    - progress\_start\_time element 601
    - ss\_exec\_time element 655
    - stmt\_elapsed\_time element 661
    - time\_completed 707
    - time\_created 707
    - time\_of\_violation 707
    - time\_started 708
    - total\_sort\_time 720
  - time zones
    - time\_zone\_disp element 708
  - timestamps
    - activate\_timestamp 308
    - db\_conn\_time 392
    - db2start\_time 392
    - last\_backup 474
    - last\_reset 475
    - lock\_wait\_start\_time 490
    - message\_time 516
    - statistics\_timestamp 658
    - status\_change\_time 660
    - stmt\_start 668
    - stmt\_stop 668
  - tokens
    - consistency\_token monitor element 373
    - corr\_token monitor element 383
  - total\_hash\_loops element 714
  - tq\_cur\_send\_spills 728
  - tq\_id\_waiting\_on 728
  - tq\_max\_send\_spills 729
  - tq\_node\_waited\_for 729
  - tq\_rows\_read 730
  - tq\_rows\_written 730
  - tq\_wait\_for\_any 731
  - transaction processing
    - client\_acctng 353
    - client\_userid 358
    - client\_wrkstnname 359
    - tpmon\_acc\_str 726
    - tpmon\_client\_userid 727
    - tpmon\_client\_wkstn 727
  - transactions
    - num\_indoubt\_trans 521
    - xid monitor element 750
  - units of work (UOW)
    - completion\_status 362
    - parent\_uow\_id 539
    - prev\_uow\_stop\_time 596
    - progress\_total\_units element 602
    - uow\_comp\_status 733
    - uow\_elapsed\_time 734
    - uow\_id 734
    - uow\_start\_time 736
    - uow\_status 737
    - uow\_stop\_time 737
  - updates
    - update\_sql\_stmts element 738
  - utilities
    - utility\_dbname 739
    - utility\_description 739
    - utility\_id 740
    - utility\_invoker\_type 740
    - utility\_priority 740
    - utility\_start\_time 741
- monitor elements (*continued*)
  - utilities (*continued*)
    - utility\_state 741
    - utility\_type 741
  - valid 741
  - wait time 237
  - wait times
    - total\_wait\_time 725
  - watermarks
    - act\_cpu\_time\_top 304
    - act\_rows\_read\_top 307
    - concurrent\_act\_top 363
    - concurrent\_connection\_top 364
    - concurrent\_wlo\_act\_top 364
    - concurrent\_wlo\_top 364
    - coord\_act\_lifetime\_top 380
    - cost\_estimate\_top 384
    - lock\_wait\_time\_top 492
    - rows\_returned\_top 627
    - temp\_tablespace\_top 703
    - uow\_total\_time\_top 738
  - workload manager
    - total queue assignments 743
    - total queue time 743
  - workloads
    - wlo\_completed\_total 744
    - workload\_id 746
    - workload\_name 746
    - workload\_occurrence\_id 747
    - workload\_occurrence\_state 748
  - XQuery
    - xquery\_stmts 750
- monitor heap
  - health indicators
    - db2.mon\_heap\_util 180
- monitor switches
  - description 111
  - setting from a client application 115
  - setting from the CLP 113
- monitoring
  - buffer pool efficiency
    - administrative views 195
  - capturing a snapshot from client applications 96
  - capturing a snapshot from the command line 93
  - capturing a snapshot using SQL 84, 92
    - with file access 88
    - with SNAP\_WRITE\_FILE 86
  - data partitions 197
  - database 3
  - database activity 190, 196
  - database events 58
    - event types 9
    - sample output 75
  - health monitor 127, 134
  - locking events 21
  - memory components 187
  - open access to monitor data
    - capturing snapshot information to a file 86
    - retrieving snapshot information from a file 88
    - SYSMON authority 83
  - runtime rollback process 196
  - snapshot
    - API request types 97
    - CLP commands 93
  - unformatted event table 13
  - with db2top 105

- monitoring elements
  - locking 26
  - unit of work 233
- most recent response time for connect monitor element 363

## N

- names
  - monitor elements
    - db\_name element 393
    - dcs\_db\_name element 397
    - service\_subclass\_name 640
    - service\_superclass\_name 640
    - work\_action\_set\_name 745
    - work\_class\_name 745
  - network time
    - monitor elements
      - max\_network\_time\_1\_ms 513
      - max\_network\_time\_100\_ms 512
      - max\_network\_time\_16\_ms 512
      - max\_network\_time\_4\_ms 513
      - max\_network\_time\_500\_ms 513
      - max\_network\_time\_gt500\_ms 514
      - network\_time\_bottom 516
      - network\_time\_top 517
  - nicknames
    - health indicator 181
    - monitor elements
      - create\_nickname element 387
      - create\_nickname\_time element 387
  - nodes
    - monitor elements
      - coord\_node monitor element 383
      - node\_number element 518
      - num\_nodes\_in\_db2\_instance 524
      - ss\_node\_number element 655
  - notices 763
  - num\_indoubt\_trans element 521
  - num\_transmissions element 525
  - num\_transmissions\_group element 526
  - numbers
    - monitor elements
      - progress\_list\_cur\_seq\_num element 601
      - ss\_number element 655

## O

- objects
  - performance on Windows 210
- OLAP functions
  - monitor elements
    - active\_olap\_funcs 308
    - olap\_func\_overflows 527
    - post\_threshold\_olap\_funcs 593
    - total\_olap\_funcs 715
- operation monitor element 664
- operations
  - monitor elements
    - direct\_read\_reqs element 404
    - direct\_read\_time element 406
    - direct\_reads element 407
    - direct\_write\_reqs element 409
    - direct\_write\_time element 410
    - direct\_writes element 412
    - stmt\_operation element 664

- optimization
  - monitor elements
    - stmt\_value\_isreopt 673
- ordering DB2 books 756
- outbound bytes received
  - monitor elements
    - max\_data\_received\_1024 502
    - max\_data\_received\_128 503
    - max\_data\_received\_16384 503
    - max\_data\_received\_2048 503
    - max\_data\_received\_256 504
    - max\_data\_received\_31999 504
    - max\_data\_received\_4096 505
    - max\_data\_received\_512 505
    - max\_data\_received\_64000 506
    - max\_data\_received\_8192 506
    - max\_data\_received\_gt64000 506
    - outbound\_bytes\_received 530
    - outbound\_bytes\_received\_bottom 531
    - outbound\_bytes\_received\_top 531
- outbound bytes sent
  - monitor elements
    - max\_data\_sent\_1024 507
    - max\_data\_sent\_128 507
    - max\_data\_sent\_16384 508
    - max\_data\_sent\_2048 508
    - max\_data\_sent\_256 508
    - max\_data\_sent\_31999 509
    - max\_data\_sent\_4096 509
    - max\_data\_sent\_512 510
    - max\_data\_sent\_64000 510
    - max\_data\_sent\_8192 511
    - max\_data\_sent\_gt64000 511
    - outbound\_bytes\_sent 531
    - outbound\_bytes\_sent\_bottom 532
    - outbound\_bytes\_sent\_top 532
- outbound communication
  - monitor elements
    - outbound\_appl\_id 529
    - outbound\_comm\_address 532
    - outbound\_comm\_protocol 533
    - outbound\_sequence\_no 533
- overflow records
  - monitor elements
    - first\_overflow\_time element 440
    - last\_over\_flow\_time element 474
    - overflow\_accesses 533
    - overflow\_creates 534

## P

- package cache
  - db.pkgcache\_hitratio 180
  - monitor elements
    - pkg\_cache\_inserts 543
    - pkg\_cache\_lookups 543
    - pkg\_cache\_num\_overflow 545
    - pkg\_cache\_size\_top 545
- packages
  - monitor elements
    - package\_name 534
    - package\_schema 535
    - package\_version\_id 535
    - stmt\_pkgcache\_id element 666
- pages
  - bp\_pages\_left\_to\_remove monitor element 348
  - data\_object\_pages monitor element 390

- pages (*continued*)
  - removing
    - bp\_pages\_left\_to\_remove monitor element 348
- parallelism
  - monitor elements
    - degree\_parallelism 401
- partial\_record monitor element 539
- partition\_number monitor element 541
- partitioned database environments
  - event monitoring 73
  - global snapshots 102
  - monitor elements
    - coord\_partition\_num 383
- partitioned tables
  - reorganizing 197
- pass-through
  - monitor elements
    - passthru\_time 541
    - passthru 541
- performance
  - information
    - displaying 209
    - enabling remote access 209
  - remote databases 211
  - resetting values 211
  - Windows
    - monitoring tools 208
    - Performance Monitor objects 210
- perspectives
  - monitoring 5, 227
- pipe event monitors
  - creating 71
  - formatting output from command line 75
  - named pipe management 72
- piped\_sorts\_accepted monitor element 542
- piped\_sorts\_requested monitor element 542
- placeholder 41
- post\_shrthreshold\_sorts monitor element 592
- prefetching
  - monitor elements
    - unread\_prefetch\_pages 733
- priv\_workspace\_num\_overflows monitor element
  - description 597
- priv\_workspace\_section\_inserts monitor element
  - description 597
- priv\_workspace\_section\_lookups monitor element
  - description 598
- priv\_workspace\_size\_top monitor element
  - description 599
- problem determination
  - information available 761
  - tutorials 761
- processes
  - monitor elements
    - agent\_pid 315
- progress\_description monitor element 600
- progress\_seq\_num monitor element 601
- progress\_start\_time monitor element 601
- progress\_work\_metric monitor element 602

## Q

- queries
  - monitor elements
    - query\_card\_estimate 603
    - query\_cost\_estimate 604
    - queue\_assignments\_total 604

- queries (*continued*)
  - monitor elements (*continued*)
    - queue\_size\_top 605
    - queue\_time\_total 605
    - select\_time 635
- quiescer
  - monitor elements
    - quiescer\_auth\_id 606
    - quiescer\_obj\_id 606
    - quiescer\_state 606
    - quiescer\_ts\_id 606

## R

- range adjustment monitor element 607
- range container monitor element 607
- range number monitor element 608
- range offset monitor element 608
- range\_num\_containers monitor element 608
- ranges
  - monitor elements
    - bottom 346
    - range\_adjustment element 607
    - range\_container\_id element 607
    - range\_end\_stripe element 607
    - range\_max\_extent element 607
    - range\_max\_page\_number element 608
    - range\_num\_containers 608
    - range\_number element 608
    - range\_offset element 608
    - range\_start\_stripe element 608
    - range\_stripe\_set\_number element 609
- real-time statistics
  - monitor elements
    - stats\_fabricate\_time 659
    - stats\_fabrications 660
- rebalancing
  - monitor elements
    - current\_extent 389
    - tablespace\_rebalancer\_extents\_processed 691
    - tablespace\_rebalancer\_extents\_remaining 691
    - tablespace\_rebalancer\_last\_extent\_moved 692
    - tablespace\_rebalancer\_mode 692
    - tablespace\_rebalancer\_priority 693
    - tablespace\_rebalancer\_restart\_time 693
    - tablespace\_rebalancer\_start\_time 694
- rebinding
  - monitor elements
    - int\_auto\_rebinds element 463
- records
  - monitor elements
    - partial\_record element 539
- recovery
  - monitor elements
    - log\_to\_redo\_for\_recovery 500
- remote
  - performance 211
- reoptimization
  - monitor elements
    - stmt\_value\_isreopt 673
- reorg\_index\_id monitor element 612
- reorganization
  - health indicators
    - db.tb\_reorg\_req 174
  - monitor elements
    - page\_reorgs element 536
    - reorg\_current\_counter element 612

- reorganization (*continued*)
  - monitor elements (*continued*)
    - reorg\_max\_counter element 613
    - reorg\_max\_phase element 613
    - reorg\_phase monitor element 613
    - reorg\_phase\_start element 614
    - reorg\_rows\_compressed 614
    - reorg\_rows\_rejected\_for\_compression 615
    - reorg\_start element 615
    - reorg\_status element 615
    - reorg\_type element 616
- reorganize phase monitor element 613
- reports
  - deadlock 23
  - lock timeout 23
  - lock wait 23
  - unit of work 43
- request identifier for sql statement monitor element 649
- requests
  - monitor elements
    - rqsts\_completed\_total 630
- response time
  - monitor elements
    - delete\_time element 402
    - host\_response\_time element 458
    - insert\_time element 462
- roll-forward recovery
  - monitor elements
    - rf\_log\_num 617
    - rf\_status 618
    - rf\_timestamp 618
    - rf\_type 618
    - tablespace\_min\_recovery\_time 686
    - ts\_name 732
- rollbacks
  - monitor elements
    - int\_deadlock\_rollbacks 464
    - int\_rollbacks 465
    - rf\_status 618
    - rollback\_sql\_stmts 619
    - rolled\_back\_agent\_id 620
    - rolled\_back\_appl\_id 620
    - rolled\_back\_participant\_no 620
    - rolled\_back\_sequence\_no 621
  - monitoring progress 196
- routines
  - monitor elements
    - routine\_id 621
- rows
  - monitor elements
    - int\_rows\_inserted 467
    - int\_rows\_updated 467
    - rows\_deleted 621
    - rows\_fetched 622
    - rows\_inserted 623
    - rows\_modified 623
    - rows\_read 624
    - rows\_returned 626
    - rows\_returned\_top 627
    - rows\_selected 628
    - rows\_updated 628
    - rows\_written 629
    - sp\_rows\_selected 648
  - rows compressed monitor element 614
  - rows rejected for compression monitor element 615
  - rows returned by stored procedures monitor element 648
  - rows selected monitor element 628

- RUNSTATS utility
  - monitor elements
    - async\_runstats 335
    - sync\_runstats 675
    - sync\_runstats\_time 676

## S

- schemas
  - tables
    - table\_schema element 679
- sections
  - monitor elements
    - appl\_section\_inserts 330
    - appl\_section\_lookups 330
    - priv\_workspace\_section\_inserts element 597
    - priv\_workspace\_section\_lookups element 598
    - section\_env 632
    - section\_number element 633
  - select SQL statements executed monitor element 634
- self-describing data stream
  - database system monitor 119
  - event monitors 76
  - snapshot monitor 103
  - system monitor switches 116
- sequences
  - monitor elements
    - progress\_seq\_num element 601
    - sequence\_no 635
    - sequence\_no\_holding\_lk 636
- servers
  - monitor elements
    - product\_name 599
    - server\_instance\_name 637
    - server\_platform 637
    - server\_prdid 637
    - server\_version 638
- service-level information
  - monitor elements
    - service\_level 639
- session authorization ID 641
- shared workspace
  - health indicators
    - db.shrworkspace\_hitratio 180
  - monitor elements
    - shr\_workspace\_num\_overflows 642
    - shr\_workspace\_section\_inserts 642
    - shr\_workspace\_section\_lookups 643
    - shr\_workspace\_size\_top 643
- snapshot monitoring
  - administrative views 193
  - API request types 97
  - capturing
    - using SQL with file access 88
  - capturing to file 86
  - CLP commands 93
  - description 83
  - interpreting output for data partitions 197
  - making snapshot data available for all users 86
  - on data partitions 197
  - on partitioned database systems 102
  - output
    - samples 99
    - self-describing data stream 103
  - request types 93
  - SQL table functions 89



- snapshot monitoring (*continued*)
  - subsections
    - subsection snapshots 101
    - using a client application 96
    - using CLP 93
    - using SQL 92
    - using SQL with direct access 84
    - with SNAP\_WRITE\_FILE 86
  - snapshot time monitor element 708
  - snapshots
    - capturing
      - using SQL with file access 88
    - capturing to file 86
    - capturing with SNAP\_WRITE\_FILE 86
    - making snapshot data available for all users 86
    - monitor elements
      - time\_stamp element 708
    - SQL table functions 89
      - using SQL with direct access 84
  - sort share heap currently allocated monitor element 647
  - Sort Share Heap High Watermark monitor element 647
  - sorting
    - health indicators
      - db2.sort\_privmem\_util 169
    - monitor elements
      - active\_sorts 308
      - db.spilled\_sorts 171
      - pipeds\_sorts\_accepted element 542
      - pipeds\_sorts\_requested element 542
      - post\_shrthreshold\_sorts 592
      - post\_threshold\_sorts element 594
      - sort\_heap\_allocated element 644
      - sort\_heap\_top monitor element 645
      - sort\_overflows element 645
      - sort\_shrheap\_allocated monitor element 647
      - sort\_shrheap\_top monitor element 647
      - total\_sorts element 721, 723
  - SQL communications area (SQLCA)
    - monitor elements
      - sqlca 651
  - SQL operations
    - monitor elements
      - elapsed\_exec\_time element 416
  - SQL requests since last commit monitor element 650
  - SQL statements
    - displaying help 757
    - monitor elements
      - ddl\_sql\_stmts element 397
      - dynamic\_sql\_stmts element 414
      - failed\_sql\_stmts element 421
      - insert\_sql\_stmts element 461
      - num\_compilation element 519
      - num\_executions element 520
      - select\_sql\_stmts element 634
      - sql\_chains 649
      - sql\_reqs\_since\_commit element 650
      - sql\_stmts 650
      - static\_sql\_stmts element 657
      - stmt\_pkgcache\_id element 666
      - stmt\_query\_id element 666
      - stmt\_sorts element 667
      - stmt\_source\_id element 668
      - stmt\_text element 669
      - stmt\_value\_data element 672
      - stmt\_value\_index element 672
      - stmt\_value\_isnull element 673
      - stmt\_value\_type element 673
    - SQL statements (*continued*)
      - monitor elements (*continued*)
        - total\_exec\_time element 713
        - uid\_sql\_stmts element 732
    - SQL table functions
      - capturing health snapshots 135
      - health monitor 183
    - sql\_chains element 649
    - sql\_stmts element 650
    - SQLTEMPSPACE activity threshold
      - monitor elements
        - sqltempespace\_threshold\_id 654
    - start stripe monitor element 608
    - statement best preparation time monitor element 596
    - statement concentrator
      - monitor elements
        - eff\_stmt\_txt 415
    - statement first use time monitor element 661
    - statement history identifier monitor element 661
    - statement history list size monitor element 662
    - statement invocation identifier monitor element 662
    - statement isolation monitor element 662
    - statement last use time monitor element 663
    - statement nesting level monitor element 664
    - statement node monitor element 664
    - statement operation monitor element 664
    - statement query identifier monitor element 666
    - statement sorts monitor element 667
    - statement source identifier monitor element 668
    - statement type monitor element 670
    - statement worst preparation time monitor element 596
  - statements
    - monitor elements
      - prep\_time\_best element 596
      - prep\_time\_worst element 596
      - stmt\_first\_use\_time element 661
      - stmt\_history\_id element 661
      - stmt\_history\_list\_size element 662
      - stmt\_invocation\_id element 662
      - stmt\_isolation element 662
      - stmt\_last\_use\_time 663
      - stmt\_nest\_level element 664
      - stmt\_node\_number element 664
      - stmt\_type element 670
  - STATEMENTS event type
    - overview 9
  - states
    - health indicators
      - db.alert\_state 173
      - db.db\_op\_status 173
      - db2.db2\_op\_status 172
      - ts.ts\_op\_status 169
  - static SQL statements attempted monitor element 657
  - statistics collection
    - health indicators
      - db.tb\_runstats\_req 174
  - status
    - monitor elements
      - appl\_status 331
      - db\_status element 394
      - db2\_status element 391
      - dcs\_app1\_status element 397
      - ss\_status element 656
    - stmt\_operation element 664
  - storage paths
    - monitor elements
      - num\_db\_storage\_paths 519

- stored procedure time monitor element 675
- stored procedures
  - monitor elements
    - stored\_proc\_time element 675
    - stored\_procs element 675
- stored procedures monitor element 675
- stripe set number monitor element 609
- stripe sets
  - monitor elements
    - container\_stripe\_set 374
- subsection execution elapsed time monitor element 655
- subsection node number monitor element 655
- subsection number monitor element 655
- subsection snapshots 101
- subsection status monitor element 656
- subsections
  - snapshot monitoring
    - subsection snapshots 101
- summary
  - health indicators 161
- SYSMON authority 83
- system monitor switches
  - description 111
  - self-describing data stream 116
  - setting from a client application 115
  - setting from the CLP 113
  - types 111
- System monitoring guide and reference
  - overview xvii

## T

- table event monitors
  - creating 61
  - table management 63
- table functions
  - monitoring 5
    - activities 6
    - data objects 7
  - perspectives 5
- table queues
  - monitor elements
    - tq\_cur\_send\_spills 728
    - tq\_id\_waiting\_on 728
    - tq\_max\_send\_spills 729
    - tq\_node\_waited\_for 729
    - tq\_rows\_read 730
    - tq\_rows\_written 730
    - tq\_tot\_send\_spills 731
    - tq\_wait\_for\_any 731
- table reorganization
  - monitor element
    - reorg\_xml\_regions\_compressed 617
    - reorg\_xml\_regions\_rejected\_for\_compression 617
  - monitor elements
    - reorg\_end element 612
- table reorganize attribute flag monitor element 616
- table reorganize completion flag monitor element 611
- table reorganize end time monitor element 612
- table reorganize phase start time monitor element 614
- table reorganize start time monitor element 615
- table reorganize status monitor element 615
- table spaces
  - health indicators
    - ts.ts\_auto\_resize\_status 166
    - ts.ts\_op\_status 169
    - ts.ts\_util 167

- table spaces (*continued*)
  - health indicators (*continued*)
    - ts.ts\_util\_auto\_resize 166
    - tsc.tscont\_op\_status 169
    - tsc.utilization 168
  - monitor elements
    - bp\_tbsp\_use\_count 348
    - index\_tbsp\_id 461
    - long\_tbsp\_id 501
    - quiescer\_ts\_id 606
    - reorg\_long\_tbsp\_id 613
    - reorg\_tbsp\_id 616
    - tablespace\_auto\_resize\_enabled 681
    - tablespace\_content\_type 682
    - tablespace\_cur\_pool\_id 682
    - tablespace\_current\_size 683
    - tablespace\_extent\_size 683
    - tablespace\_free\_pages 683
    - tablespace\_id 684
    - tablespace\_increase\_size 684
    - tablespace\_increase\_size\_percent 685
    - tablespace\_initial\_size 685
    - tablespace\_last\_resize\_failed 685
    - tablespace\_last\_resize\_time 686
    - tablespace\_max\_size 686
    - tablespace\_min\_recovery\_time 686
    - tablespace\_name 687
    - tablespace\_next\_pool\_id 688
    - tablespace\_num\_containers 688
    - tablespace\_num\_quiescers 688
    - tablespace\_num\_ranges 689
    - tablespace\_page\_size 689
    - tablespace\_page\_top 689
    - tablespace\_pending\_free\_pages 690
    - tablespace\_prefetch\_size 690
    - tablespace\_rebalancer\_extents\_processed 691
    - tablespace\_rebalancer\_extents\_remaining 691
    - tablespace\_rebalancer\_last\_extent\_moved 692
    - tablespace\_rebalancer\_mode 692
    - tablespace\_rebalancer\_priority 693
    - tablespace\_rebalancer\_restart\_time 693
    - tablespace\_rebalancer\_start\_time 694
    - tablespace\_state 694
    - tablespace\_state\_change\_object\_id 695
    - tablespace\_state\_change\_ts\_id 695
    - tablespace\_total\_pages 696
    - tablespace\_type 696
    - tablespace\_usable\_pages 697
    - tablespace\_used\_pages 697
    - tablespace\_using\_auto\_storage 698
    - tbsp\_max\_page\_top 698
    - ts\_name 732
- tables
  - monitor elements
    - tab\_file\_id 677
    - tab\_type 677
    - table\_file\_id element 678
    - table\_name element 678
    - table\_scans 679
    - table\_schema element 679
    - table\_type element 681
- TABLES event type
  - overview 9
- TABLESPACES event type
  - overview 9



- TCP/IP
  - monitor elements
    - tcpip\_sends\_total 703
- terms and conditions
  - use of publications 762
- territory codes
  - monitor elements
    - territory\_code 704
- threads
  - monitor elements
    - agent\_pid 315
- thresholds
  - monitor elements
    - num\_threshold\_violations 525
    - sqltemp space\_threshold\_id 654
    - threshold\_action 704
    - threshold\_domain 705
    - threshold\_maxvalue 705
    - threshold\_name 705
    - threshold\_predicate 706
    - threshold\_queuesize 706
    - thresholdid 706
  - threshold-based health indicators 127, 159
- time
  - monitor elements
    - prefetch\_wait\_time element 595
    - prep\_time 595
    - progress\_start\_time element 601
    - ss\_exec\_time element 655
    - stmt\_elapsed\_time element 661
    - time\_completed 707
    - time\_created 707
    - time\_of\_violation 707
    - time\_started 708
    - total\_sort\_time 720
  - time waited for prefetch monitor element 595
  - time zone displacement monitor element 708
- time zones
  - monitor elements
    - time\_zone\_disp element 708
- timestamps
  - monitor elements
    - activate\_timestamp 308
    - db\_conn\_time 392
    - db2start\_time 392
    - last\_backup 474
    - last\_reset 475
    - lock\_wait\_start\_time 490
    - message\_time 516
    - prev\_uow\_stop\_time 596
    - statistics\_timestamp 658
    - status\_change\_time 660
    - stmt\_start 668
    - stmt\_stop 668
    - uow\_start\_time 736
    - uow\_stop\_time 737
- tokens
  - monitor elements
    - consistency\_token monitor element 373
    - corr\_token monitor element 383
- total fcm buffers received monitor element 711
- total fcm buffers sent monitor element 711
- total hash loops monitor element 714
- total log available monitor element 714
- total log space used monitor element 715
- total number of pages in object monitor element 613
- total progress work units monitor element 602
- total sort time monitor element 720
- total sorts monitor element 721, 723
- transaction processing monitors
  - monitor elements
    - client\_acctng 353
    - client\_applname 354
    - client\_userid 358
    - client\_wrkstnname 359
    - tpmon\_acc\_str 726
    - tpmon\_client\_app 726
    - tpmon\_client\_userid 727
    - tpmon\_client\_wkstn 727
- transactions
  - monitor elements
    - num\_indoubt\_trans 521
    - xid monitor element 750
- TRANSACTIONS event type
  - overview 9
- troubleshooting
  - online information 761
  - tutorials 761
- tutorials
  - problem determination 761
  - troubleshooting 761
  - Visual Explain 761
- type at monitored (server) node monitor element 636

## U

- unformatted event table
  - db2evmonfmt Java-based tool for parsing data 16
- unit of work event monitor reports 43
- unit of work monitor element
  - written to relational table 48
- unit of work monitor event
  - written to XML 44
- units of work (UOW)
  - monitor elements
    - completion\_status 362
    - parent\_uow\_id 539
    - prev\_uow\_stop\_time 596
    - progress\_total\_units element 602
    - uow\_comp\_status 733
    - uow\_elapsed\_time 734
    - uow\_id 734
    - uow\_lock\_wait\_time 735
    - uow\_log\_space\_used 735
    - uow\_start\_time 736
    - uow\_status 737
    - uow\_stop\_time 737
  - update response time monitor element 738
  - update\_time element 738
  - update/insert/delete SQL statements executed monitor element 732
- updates
  - DB2 Information Center 758, 759
  - monitor elements
    - update\_sql\_stmts element 738
- updates monitor element 738
- user authorization level monitor element
  - authorizations
    - authority\_lvl element 342
- utilities
  - monitor elements
    - utility\_dbname 739
    - utility\_description 739
    - utility\_id 740

utilities (*continued*)  
  monitor elements (*continued*)  
    utility\_invoker\_type 740  
    utility\_priority 740  
    utility\_start\_time 741  
    utility\_state 741  
    utility\_type 741

xquery\_stmts monitor element 750

## V

value data monitor element 672  
value has null value monitor element 673  
value index monitor element 672  
value type monitor element 673  
version monitor element 742  
version of monitor data monitor element 742  
Visual Explain  
  tutorial 761

## W

wait times  
  monitor elements  
    total\_wait\_time 725  
watermark monitor elements  
  act\_cpu\_time\_top 304  
  act\_rows\_read\_top 307  
  concurrent\_connection\_top 364  
  concurrent\_wlo\_act\_top 364  
  concurrent\_wlo\_top 364  
  coord\_act\_lifetime\_top 380  
  cost\_estimate\_top 384  
  lock\_wait\_time\_top 492  
  uow\_total\_time\_top 738  
watermarks  
  monitor elements  
    concurrent\_act\_top 363  
    rows\_returned\_top 627  
    temp\_tablespace\_top 703  
Windows Management Instrumentation (WMI)  
  DB2 database system integration 207  
  description 206  
Windows operating systems  
  Performance Monitor  
    description 208  
    registering DB2 208  
WMI (Windows Management Instrumentation)  
  see Windows Management Instrumentation (WMI) 206  
workload manager  
  monitor elements  
    total\_queue\_assignments 743  
    total\_queue\_time 743  
workloads  
  monitor elements  
    wlo\_completed\_total 744  
    workload\_id 746  
    workload\_name 746  
    workload\_occurrence\_id 747  
    workload\_occurrence\_state 748  
write-to-table event monitors  
  buffering 71

## X

XDA Object Pages monitor element 749  
xda\_object\_pages monitor element 749







Printed in USA

SC27-2458-00



Spine information:

IBM DB2 9.7 for Linux, UNIX, and Windows

Database Monitoring Guide and Reference

