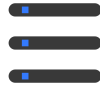


Sharing IBM i Knowledge

# System Administration Modernisation



Getting started in the world of SQL performance

DATE

14 June 2026

Host

[Rudi van Helvoirt](#)



” You only know  
what it is, when it  
is no longer there.. ”

*The ultimate definition of an invisible foundation.*



```

Examples
daily SQL Plan Cache management
Daily SQL Plan Cache management
-- category: Db2 for i Services
-- description: Daily SQL Plan Cache management
CL: CRTLIB SNAPSHOTS;
CL: CRTLIB EVENTMONS;
-- Purpose: This procedure captures detail on SQL queries.
-- 1) The 100 most expensive SQL queries are captured into a SQL Plan Cache Snapshot named SNAP.
-- 2) An SQL Plan Cache Event Monitor is started using a name SNAPSHOTS/EVT<julian-date>. The p
-- 3) For both 1 & 2, only the 14 most recent days are kept online.
-- 4) For both 1 & 2, the new monitor and snap shot are imported into the IBM i Access Client S
CREATE OR REPLACE PROCEDURE SNAPSHOTS.DAILY_PC_MANAGEMENT()
LANGUAGE SQL
BEGIN
  DECLARE not_found CONDITION FOR '02000';
  DECLARE SNAP_NAME CHAR(10);
  DECLARE OLDEST_SNAP_NAME CHAR(10);
  DECLARE SNAP_COMMENT VARCHAR(100);
  DECLARE EVENT_MONITOR_NAME CHAR(10);
  DECLARE YESTERDAY_EVENT_MONITOR_NAME CHAR(10);
  DECLARE OLDEST_EVENT_MONITOR_NAME CHAR(10);
  DECLARE OLD_EVENT_MONITOR_ID CHAR(10);
  DECLARE v_not_found BIGINT DEFAULT 0;

  -- A Julian date is the integer value representing a number of days
  -- from January 1, 4713 B.C. (the start of the Julian calendar) to
  -- the date specified in the argument.
  SET SNAP_NAME = 'SNP' || CONCAT JULIAN_DAY(current date);
  SET OLDEST_SNAP_NAME = 'SNP' || CONCAT JULIAN_DAY(current date - 14 days);
  SET EVENT_MONITOR_NAME = 'EVT' || CONCAT JULIAN_DAY(current date);
  SET OLDEST_EVENT_MONITOR_NAME = 'EVT' || CONCAT JULIAN_DAY(current date - 14 days);
  SET YESTERDAY_EVENT_MONITOR_NAME = 'EVT' || CONCAT JULIAN_DAY(current date - 1 day);

  -----
  -- Process the Top 100 most expensive queries
  -- Capture the topn queries and import the snapshot
  CALL QSYS2.DUMP_PLAN_CACHE_topN('SNAPSHOTS', SNAP_NAME, 100);

  -- Remove the oldest TOPN snapshot
  BEGIN
  DECLARE CONTINUE HANDLER FOR not_found
  
```

Where my journey started

## Entering the world of SQL performance

### ✓ Presentation given by Scott Forstie

No need to explain.

It is what we have been using since the day the AS/400 saw the light, right?

### ✓ IBM i Access Client Solutions

Insert from examples

### ✓ IBM i services (SQL)

Making the move from CL command & 5250 emulation to Run SQL Scripts

# Daily Plan Cache Management

We did change this:

```

24 -- A Julian date is the integer value representing a number of days
25 -- from January 1, 4713 B.C. (the start of the Julian calendar) to
26 -- the date specified in the argument.
27 SET SNAP_NAME = 'SNP' CONCAT JULIAN_DAY(current date);
28 SET OLDEST_SNAP_NAME = 'SNP' CONCAT JULIAN_DAY(current date - 14 days);
29 SET EVENT_MONITOR_NAME = 'EVT' CONCAT JULIAN_DAY(current date);
30 SET OLDEST_EVENT_MONITOR_NAME = 'EVT' CONCAT JULIAN_DAY(current date - 14 days);
31 SET YESTERDAY_EVENT_MONITOR_NAME = 'EVT' CONCAT JULIAN_DAY(current date - 1 day);

```

Into this:

```

32
33 SET SNAP_NAME = TRIM ( PREFIX ) || '_' CONCAT SUBSTR ( REPLACE ( CHAR ( CURRENT DATE , ISO ) , '-' , '' ) , 3 , 6 );
34 SET OLDEST_SNAP_NAME = TRIM ( PREFIX ) || '_' CONCAT SUBSTR ( REPLACE ( CHAR ( CURRENT DATE - 14 DAYS , ISO ) , '-' , '' ) , 3 , 6 );
35 SET EVENT_MONITOR_NAME = TRIM ( PREFIX ) || '_' CONCAT SUBSTR ( REPLACE ( CHAR ( CURRENT DATE , ISO ) , '-' , '' ) , 3 , 6 );
36 SET OLDEST_EVENT_MONITOR_NAME = TRIM ( PREFIX ) || '_' CONCAT SUBSTR ( REPLACE ( CHAR ( CURRENT DATE - 7 DAYS , ISO ) , '-' , '' ) , 3 , 6 );
37 SET YESTERDAY_EVENT_MONITOR_NAME = TRIM ( PREFIX ) || '_' CONCAT SUBSTR ( REPLACE ( CHAR ( CURRENT DATE - 1 DAYS , ISO ) , '-' , '' ) , 3 , 6 );
38 -----

```

# Daily Plan Cache Management

Resulting in:

Prod Plan Cache Snapshots

Name	Schema	Table	Created By	Date Created
SNAPSHOTS ASP_2605110511234500	SNAPSHOTS	ASP_260511	RUDI	05/11/2026 11:45:00 PM
SNAPSHOTS ASP_2605120512234500	SNAPSHOTS	ASP_260512	RUDI	05/12/2026 11:45:00 PM
SNAPSHOTS ASP_2605130513234500	SNAPSHOTS	ASP_260513	RUDI	05/13/2026 11:45:00 PM
SNAPSHOTS ASP_2605140514234500	SNAPSHOTS	ASP_260514	RUDI	05/14/2026 11:45:00 PM
SNAPSHOTS ASP_2605150515234500	SNAPSHOTS	ASP_260515	RUDI	05/15/2026 11:45:00 PM
SNAPSHOTS ASP_2605160516234501	SNAPSHOTS	ASP_260516	RUDI	05/16/2026 11:45:01 PM
SNAPSHOTS ASP_2605170517234501	SNAPSHOTS	ASP_260517	RUDI	05/17/2026 11:45:01 PM
SNAPSHOTS ASP_2605180518234500	SNAPSHOTS	ASP_260518	RUDI	05/18/2026 11:45:00 PM
SNAPSHOTS ASP_2605190519234500	SNAPSHOTS	ASP_260519	RUDI	05/19/2026 11:45:00 PM
SNAPSHOTS ASP_2605200520234500	SNAPSHOTS	ASP_260520	RUDI	05/20/2026 11:45:00 PM
SNAPSHOTS ASP_2605210521234501	SNAPSHOTS	ASP_260521	RUDI	05/21/2026 11:45:01 PM
SNAPSHOTS ASP_2605220522234500	SNAPSHOTS	ASP_260522	RUDI	05/22/2026 11:45:00 PM
SNAPSHOTS ASP_2605230523234501	SNAPSHOTS	ASP_260523	RUDI	05/23/2026 11:45:01 PM
SNAPSHOTS ASP_2605240524234501	SNAPSHOTS	ASP_260524	RUDI	05/24/2026 11:45:01 PM
SNAPSHOTS SYS_2605110511234501	SNAPSHOTS	SYS_260511	HA	05/11/2026 11:45:01 PM
SNAPSHOTS SYS_2605120512234501	SNAPSHOTS	SYS_260512	HA	05/12/2026 11:45:01 PM
SNAPSHOTS SYS_2605130513234501	SNAPSHOTS	SYS_260513	HA	05/13/2026 11:45:01 PM
SNAPSHOTS SYS_2605140514234501	SNAPSHOTS	SYS_260514	HA	05/14/2026 11:45:01 PM
SNAPSHOTS SYS_2605150515234501	SNAPSHOTS	SYS_260515	HA	05/15/2026 11:45:01 PM
SNAPSHOTS SYS_2605160516234502	SNAPSHOTS	SYS_260516	HA	05/16/2026 11:45:02 PM
SNAPSHOTS SYS_2605170517234501	SNAPSHOTS	SYS_260517	HA	05/17/2026 11:45:01 PM
SNAPSHOTS SYS_2605180518234501	SNAPSHOTS	SYS_260518	HA	05/18/2026 11:45:01 PM
SNAPSHOTS SYS_2605190519234501	SNAPSHOTS	SYS_260519	HA	05/19/2026 11:45:01 PM
SNAPSHOTS SYS_2605200520234501	SNAPSHOTS	SYS_260520	HA	05/20/2026 11:45:01 PM
SNAPSHOTS SYS_2605210521234502	SNAPSHOTS	SYS_260521	HA	05/21/2026 11:45:02 PM
SNAPSHOTS SYS_2605220522234501	SNAPSHOTS	SYS_260522	HA	05/22/2026 11:45:01 PM
SNAPSHOTS SYS_2605230523234501	SNAPSHOTS	SYS_260523	HA	05/23/2026 11:45:01 PM
SNAPSHOTS SYS_2605240524234502	SNAPSHOTS	SYS_260524	HA	05/24/2026 11:45:02 PM

Prod Plan Cache Event Monitors

Name	Schema	Table	Created By	Status	Date Created
EVENTMONS SYS_260518PLANC00214	EVENTMONS	SYS_260518	HA	Ended	05/18/2026 11:45:07 PM
EVENTMONS SYS_260519PLANC00215	EVENTMONS	SYS_260519	HA	Ended	05/19/2026 11:45:10 PM
EVENTMONS SYS_260520PLANC00216	EVENTMONS	SYS_260520	HA	Ended	05/20/2026 11:45:12 PM
EVENTMONS SYS_260521PLANC00217	EVENTMONS	SYS_260521	HA	Ended	05/21/2026 11:45:12 PM
EVENTMONS SYS_260522PLANC00218	EVENTMONS	SYS_260522	HA	Ended	05/22/2026 11:45:07 PM
EVENTMONS SYS_260523PLANC00219	EVENTMONS	SYS_260523	HA	Ended	05/23/2026 11:45:13 PM
EVENTMONS SYS_260524PLANC00220	EVENTMONS	SYS_260524	HA	Started	05/24/2026 11:45:07 PM

# Daily Plan Cache Management

For iASP handling an extra parameter was added:

For \*SYSBAS Plan Cache Dumps

```
RUNSQL SQL('CALL SNAPSHOTS.DAILY_PC_MANAGEMENT("SYS"') COMMIT(*NONE) NAMING(*SQL)
```

For \*iASP Plan Cache Dumps

```
RUNSQL SQL('CALL SNAPSHOTS.DAILY_PC_MANAGEMENT("ASP"') COMMIT(*NONE) NAMING(*SQL)
```

In the code this section was added:

```
52 -- Only do the event monitor once, that is when run in *SYSBAS
53 IF PREFIX = 'SYS'
54 THEN |
55 BEGIN
56 -- If we found yesterdays event monitor, end it
57 BEGIN
58 DECLARE CONTINUE HANDLER FOR SQLEXCEPTION
59 SET V_NOT_FOUND = 1 ;
60 CALL QSYS2 . END_PLAN_CACHE_EVENT_MONITOR ( YESTERDAY_EVENT_MONITOR_NAME ) ;
61 END ;
```

# Plan Cache Event Monitor

Access Plan Rebuild Reason:

Name	Schema	Table	Created By	Status	Date Created
EVENTMONS SYS_260517PLANC00019	EVENTMONS	SYS_260517	RUDI	Ended	05/17/2026 11:55:03 PM
EVENTMONS SYS_260518PLANC00020	EVENTMONS	SYS_260518	RUDI	Ended	05/18/2026 11:55:06 PM
EVENTMONS SYS_260519PLANC00021	EVENTMONS	SYS_260519	RUDI	Ended	05/19/2026 11:55:05 PM
EVENTMONS SYS_260520PLANC00022	EVENTMONS	SYS_260520	RUDI	Ended	05/20/2026 11:55:05 PM
EVENTMONS SYS_260521PLANC00023	EVENTMONS	SYS_260521	RUDI	Ended	05/21/2026 11:55:08 PM
EVENTMONS SYS_260522PLANC00024	EVENTMONS	SYS_260522	RUDI	Ended	05/22/2026 11:55:04 PM
EVENTMONS SYS_260523PLANC00025	EVENTMONS	SYS_260523	RUDI	Ended	05/23/2026 11:55:04 PM
EVENTMONS SYS_260524PLANC00026	EVENTMONS	SYS_260524	RUDI	Ended	05/24/2026 11:55:04 PM
EVENTMONS SYS_260525PLANC00027	EVENTMONS	SYS_260525	RUDI	Ended	05/25/2026 11:55:04 PM
EVENTMONS SYS_260526PLANC00028	EVENTMONS	SYS_260526	RUDI	Ended	
EVENTMONS SYS_260527PLANC00029	EVENTMONS	SYS_260527	RUDI	Ended	
EVENTMONS SYS_260528PLANC00030	EVENTMONS	SYS_260528	RUDI	Ended	
EVENTMONS SYS_260529PLANC00031	EVENTMONS	SYS_260529	RUDI	Ended	
EVENTMONS SYS_260530PLANC00032	EVENTMONS	SYS_260530	RUDI	Started	

- Analyze
- Show Statements
- Compare
- Comments...
- End
- Delete...
- Rename...
- Properties

Metric	Value	Reports
SQL Statements	6,172	< Select a report >
Users	4	< Select a report >
Jobs	37	< Select a report >
Threads	37	
Average Table Rows	766.703	
Average Rows Returned	1.035	
Average Runtime	0.000226	
Average Parallel Degree Used	1.00	
Maximum Parallel Degree	1.00	
SQE	6,172	< Select a report >
CQE	0	
System Naming	52	< Select a report >
SQL Naming	102	< Select a report >
Unique Open Statements	80	< Select a report >
Full Opens	6,056	< Select a report >
Pseudo Opens	116	< Select a report >
Table Scans	1,870	< Select a report >
Average MQTs Used	0.000	
Average Indexes Used	0.945	< Select a report >
Full Indexes Created	42	< Select a report >
Sparse Indexes Created	0	
Index From Index Created	19	< Select a report >
Index Creates Advised	5,970	< Select a report >
Advised Statistics	99	< Select a report >
Temporary Tables	3	< Select a report >
Sorts	1,466	< Select a report >
Access Plans Rebuilt	157	< Select a report >
Sort Sequence	0	< Select a report >
Call Statements	0	Access Plan Rebuild Summary
Error	0	Access Plan Rebuild Statements

# Plan Cache Event Monitor

EVENTMONS SYS\_260527PLANC00029 - Access Plan Rebuild Statements


File Edit View Actions Help

Start Time	Last Access Plan Rebuilt	Rebuild Deferred	Access Plan Rebuild Reason	Rebuild Subcode	Original Rebuild Reason	Original Rebuild Subcode	Access Plan Save Reason	Acc Pla Ler
2026-04-30 00:04:48.012430	2026-04-30 00:04:47.614001	No	New access plan	0000		0000		
2026-05-02 00:01:50.225397	2026-05-02 00:01:50.126463	No	New access plan	0000		0000		
2026-05-02 00:01:52.385464	2026-05-02 00:01:51.795737	No	New access plan	0000		0000		
2026-05-02 00:03:26.996089	2026-05-02 00:03:26.937573	No	New access plan	0000		0000		
2026-05-03 00:02:00.152706	2026-05-03 00:02:00.002883	No	New access plan	0000		0000		
2026-05-03 00:03:48.703556	2026-05-03 00:03:23.733609	No	New access plan	0000		0000		
2026-05-03 08:00:27.372213	2026-04-29 23:12:52.023679	No	New access plan	0000		0000		
2026-05-03 08:00:44.673759	2026-05-03 08:00:44.653600	No	System programming change	000A		0000		
2026-05-03 08:01:29.165326	2026-05-03 08:01:29.153531	No	System programming change	000A		0000		
2026-05-05 00:01:52.142694	2026-05-05 00:01:52.052709	No	New access plan	0000		0000		
2026-05-05 00:01:52.543667	2026-05-05 00:01:52.503284	No	New access plan	0000		0000		
2026-05-05 00:01:56.063791	2026-05-05 00:01:55.813478	No	New access plan	0000		0000		
2026-05-05 00:02:00.366530	2026-05-05 00:01:56.343255	No	New access plan	0000		0000		
2026-05-05 00:03:40.024234	2026-05-05 00:03:39.968572	No	New access plan	0000		0000		
2026-05-06 00:04:18.689495	2026-05-06 00:04:18.568682	No	New access plan	0000		0000		
2026-05-06 13:03:26.383766	2026-05-06 13:03:26.371021	No	Significant change in the number of rows	0004		0000		
2026-05-06 13:04:42.936553	2026-05-06 13:04:42.796749	No	New access plan	0000		0000		
2026-05-06 13:04:57.498211	2026-05-06 13:04:57.477354	No	New access plan	0000		0000		
2026-05-06 14:58:29.768475	2026-05-06 14:58:29.626569	No	New access plan	0000		0000		
2026-05-06 14:58:38.020180	2026-05-06 13:04:57.138408	No	Significant change in the number of rows	0004		0000		
2026-05-06 15:07:58.179463	2026-05-06 15:07:46.846805	No	New access plan	0000		0000		
2026-05-07 00:01:51.475986	2026-05-07 00:01:51.434417	No	New access plan	0000		0000		
2026-05-07 00:01:52.323943	2026-05-07 00:01:52.275534	No	New access plan	0000		0000		
2026-05-07 00:01:52.805226	2026-05-07 00:01:52.675520	No	New access plan	0000		0000		
2026-05-08 00:03:22.170658	2026-05-08 00:02:36.011108	No	New access plan	0000		0000		
2026-05-08 00:03:22.290221	2026-05-08 00:03:22.231101	No	New access plan	0000		0000		
2026-05-08 00:03:22.339082	2026-05-08 00:03:22.318264	No	New access plan	0000		0000		
2026-05-09 00:04:22.858166	2026-05-09 00:02:56.617015	No	New access plan	0000		0000		

100 rows retrieved (more data available).

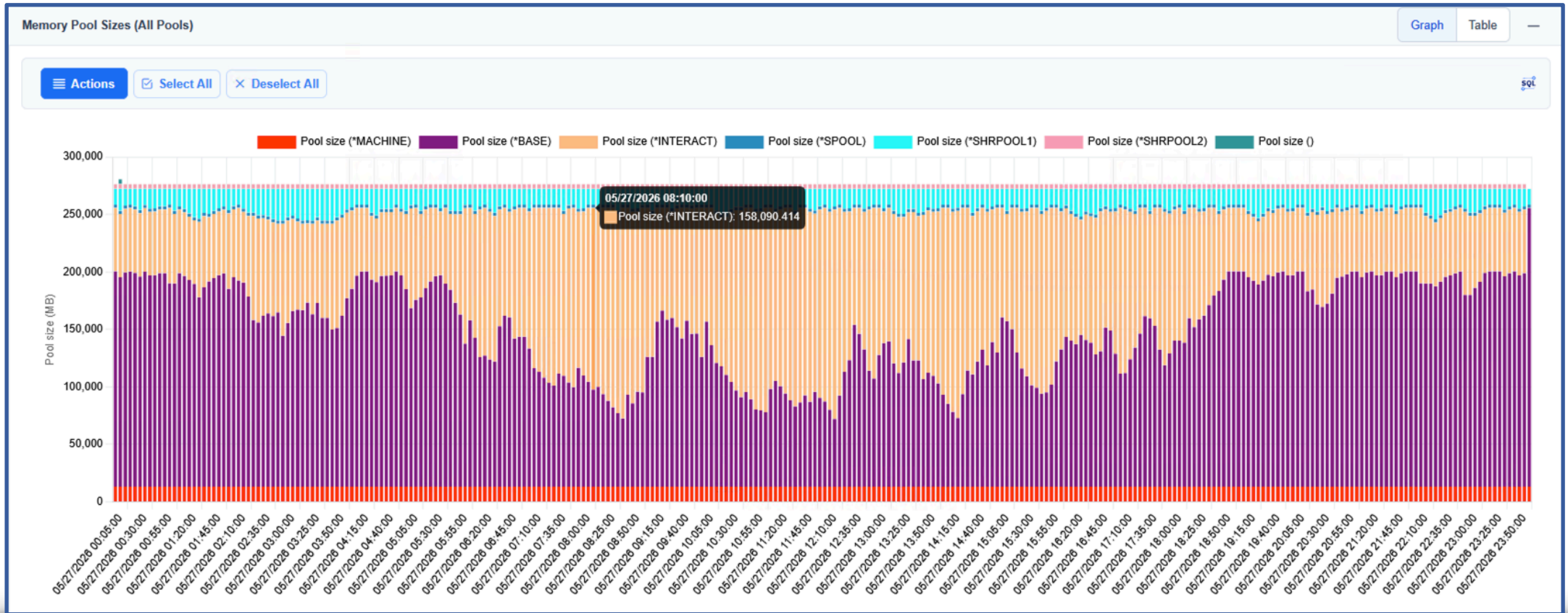
# Plan Cache Event Monitor

## Database monitor view 3006 - Access Plan Rebuild

Reason_Code	QQRCOD	Description
		<ul style="list-style-type: none"> <li>- A3 - Access plan was built to use a non-reusable Open Data Path (ODP) and ODP for this call.</li> <li>- A4 - The number of rows in the table has changed by more than 10% since</li> <li>- A5 - A new index exists over one of the tables in the query</li> <li>- A6 - An index that was used for this access plan no longer exists or is no longer</li> <li>- A7 - IBM i Query requires the access plan to be rebuilt because of system p</li> <li>- A8 - The CCSID of the current job is different than the CCSID of the job that</li> <li>- A9 - The value of one or more of the following is different for the current job this access plan:               <ul style="list-style-type: none"> <li>▪ date format</li> <li>▪ date separator</li> <li>▪ time format</li> <li>▪ time separator.</li> </ul> </li> <li>- AA - The sort sequence table specified is different than the sort sequence table that was created.</li> <li>- AB - Storage pool changed. </li> <li>- AC - The system feature Db2® multisystem has been installed or removed.</li> <li>- AD - The value of the degree query attribute has changed.</li> <li>- AE - A view is either being opened by a high level language or a view is being</li> </ul>

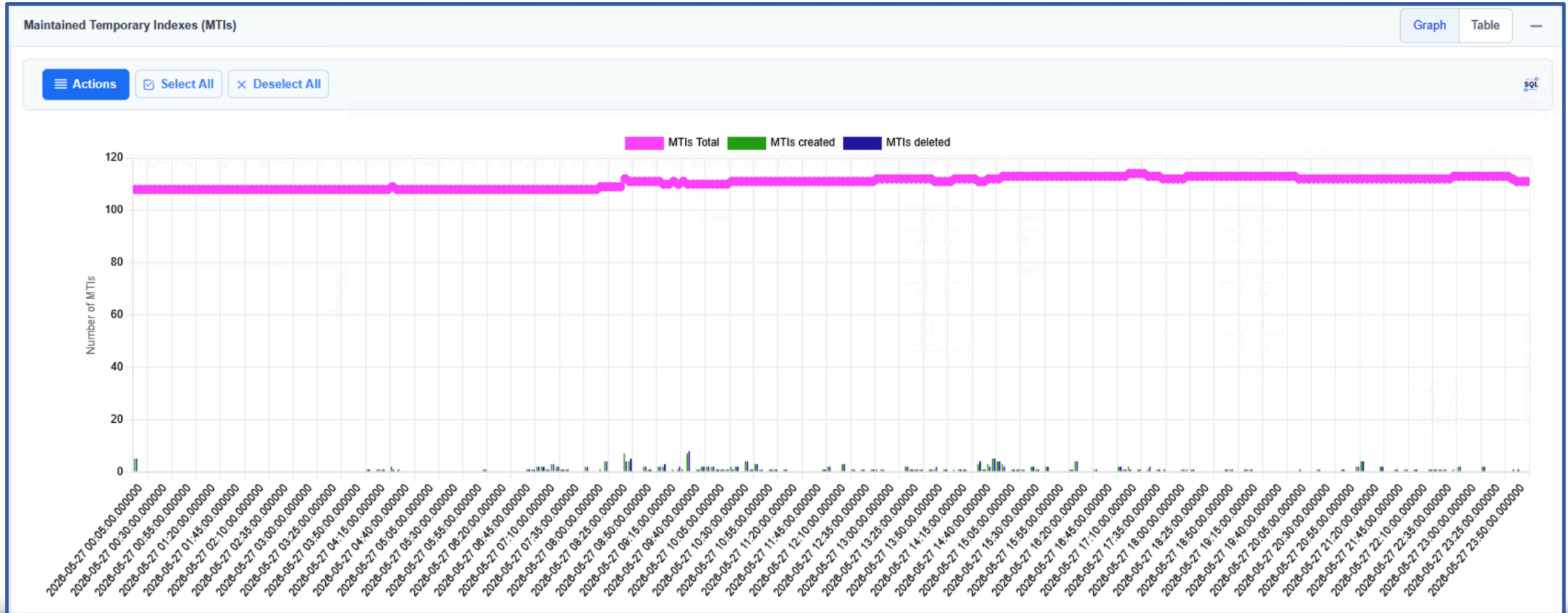
# Plan Cache Event Monitor

## Memory Pool Sizes (All Pools)



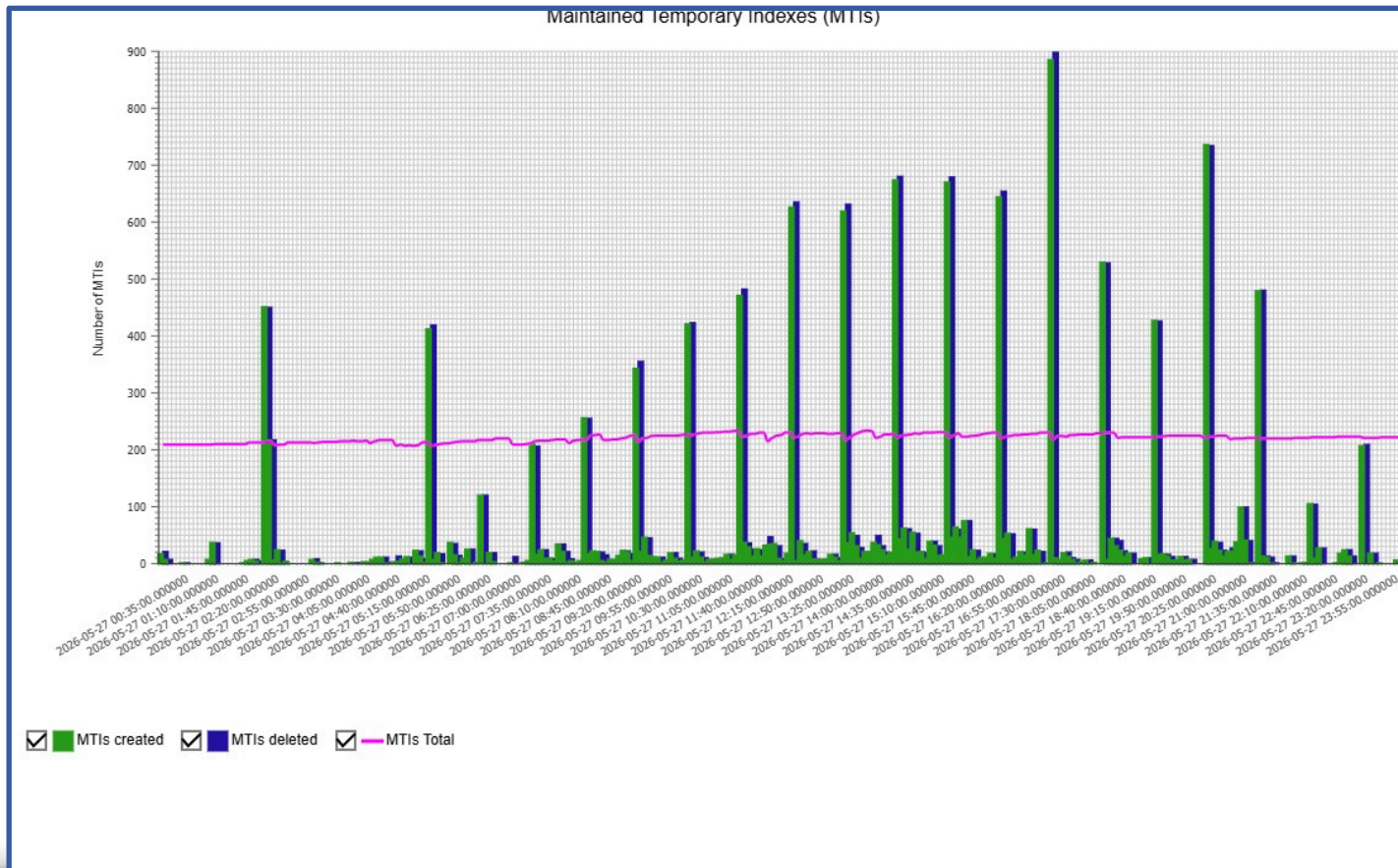
# Plan Cache Event Monitor

## Maintained Temporary Indexes (MTIs)



# Plan Cache Event Monitor

Maintained Temporary Indexes (MTIs) - but what if your graph looks like this?



# Plan Cache Event Monitor

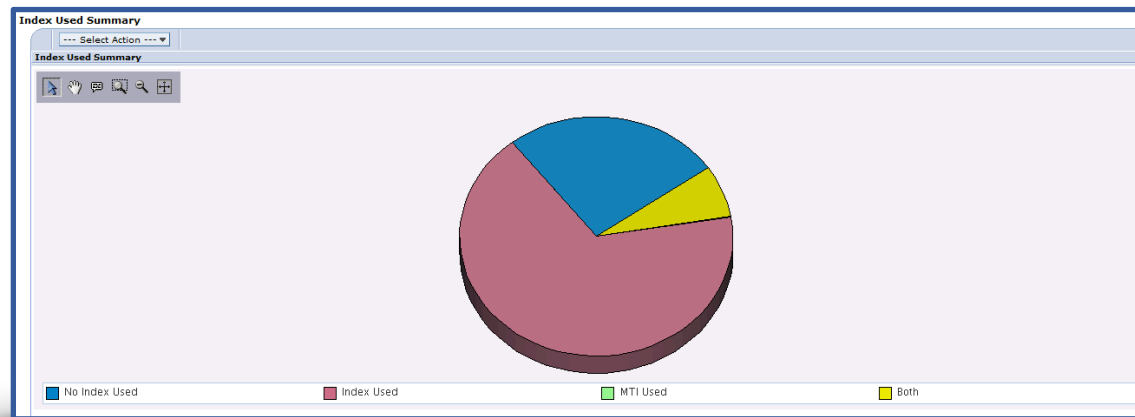
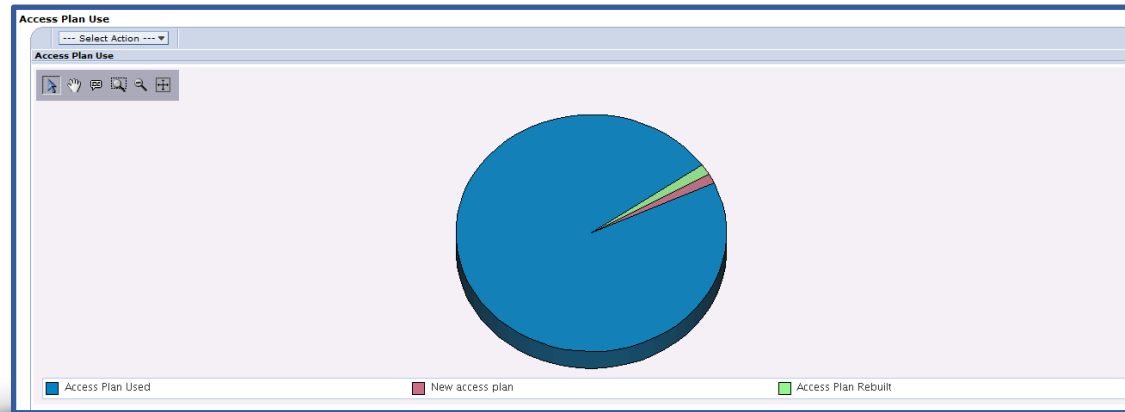
Maintained Temporary Indexes (MTIs) - With the Plan Cache Event Monitor only showing a dozen of Plans being removed during that time window?

In that case the answer can be found in the Audit Journal DO (Delete Object) entries:

748	11A	Object was <u>deleted</u> (not under commitment control)	AT▶O233698	*FILE	PF-DTA	*SYSBAS	1		
749	11A	Object was <u>deleted</u> (not under commitment control)	AT▶O233696	*FILE	PF-DTA	*SYSBAS	1		
750	11A	Object was <u>deleted</u> (not under commitment control)	AT▶O233700	*FILE	PF-DTA	*SYSBAS	1		
751	11A	Object was <u>deleted</u> (not under commitment control)	AT▶O233699	*FILE	PF-DTA	*SYSBAS	1		
752	11A	Object was <u>deleted</u> (not under commitment control)	AT▶O233701	*FILE	PF-DTA	*SYSBAS	1		
753	12A	Object was <u>deleted</u> (not under commitment control)	AT▶O233702	*FILE	PF-DTA	*SYSBAS	1		
754	12A	Object was <u>deleted</u> (not under commitment control)	AT▶D3440806	*FILE	PF-DTA	*SYSBAS	1		
755	16A	Object was <u>deleted</u> (not under commitment control)	AT▶O233703	*FILE	PF-DTA	*SYSBAS	1		
756	17A	Object was <u>deleted</u> (not under commitment control)	AT▶O233704	*FILE	PF-DTA	*SYSBAS	1		
757	17A	Object was <u>deleted</u> (not under commitment control)	AT▶O233705	*FILE	PF-DTA	*SYSBAS	1		
758	19A	Object was <u>deleted</u> (not under commitment control)	GW▶ROI8787	*FILE	PF-DTA	*SYSBAS	1		
759	7A	Object was <u>deleted</u> (not under commitment control)	AT▶O233706	*FILE	PF-DTA	*SYSBAS	1		
760	8A	Object was <u>deleted</u> (not under commitment control)	AT▶O233707	*FILE	PF-DTA	*SYSBAS	1		
761	7A	Object was <u>deleted</u> (not under commitment control)	AT▶Z3440816	*FILE	PF-DTA	*SYSBAS	1		
762	7A	Object was <u>deleted</u> (not under commitment control)	AT▶Z3440817	*FILE	PF-DTA	*SYSBAS	1		

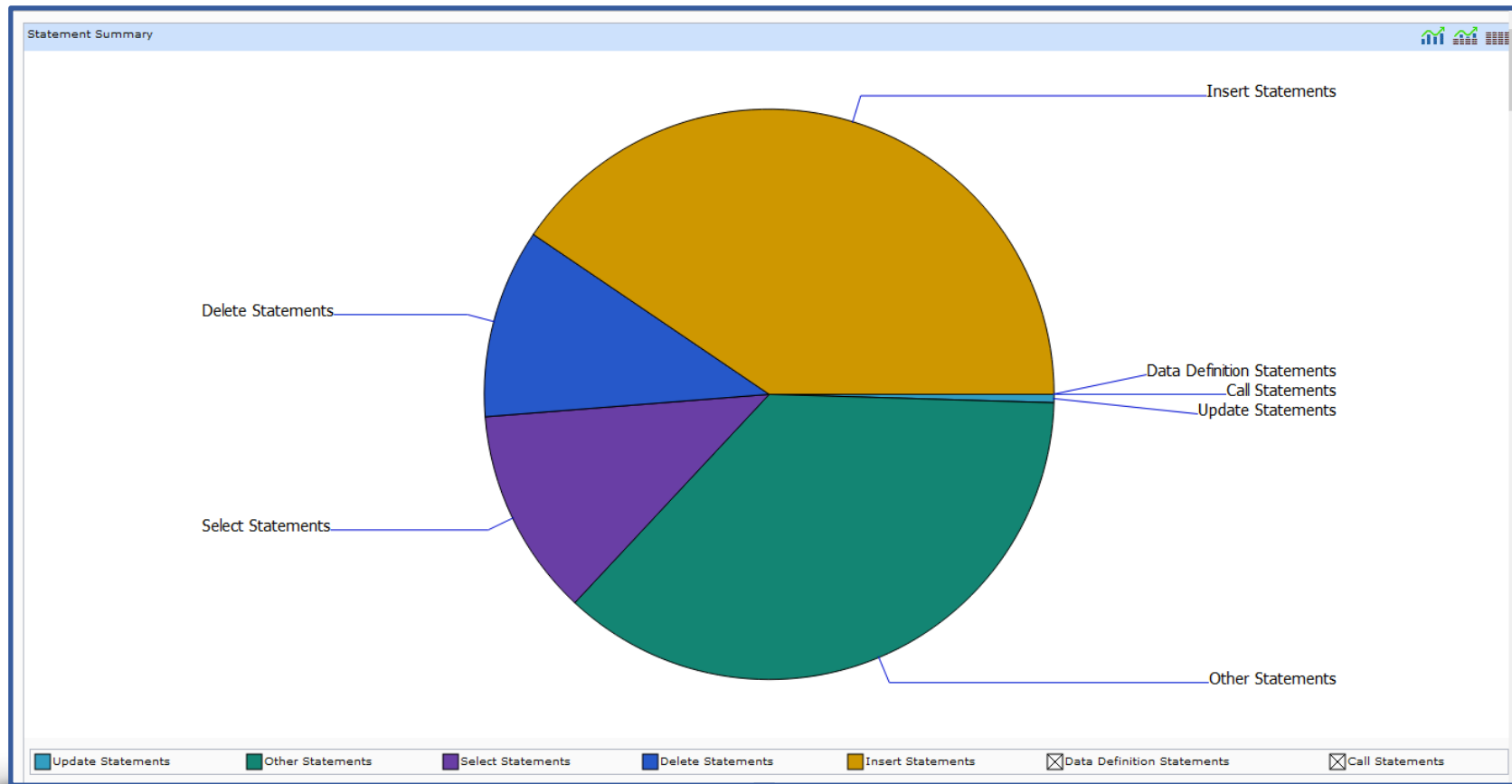
# Plan Cache Snapshots

Plan Cache Snapshots Navigator for i - will be back soon 😊



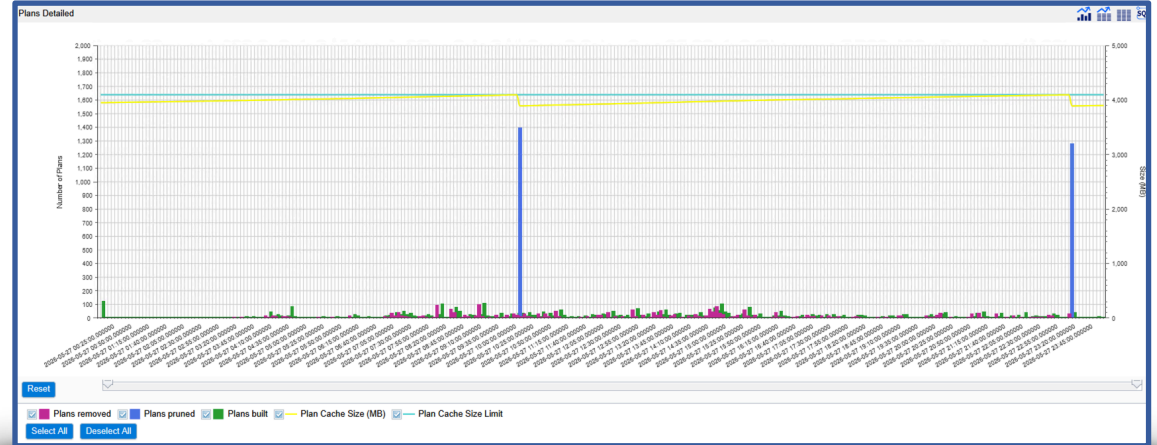
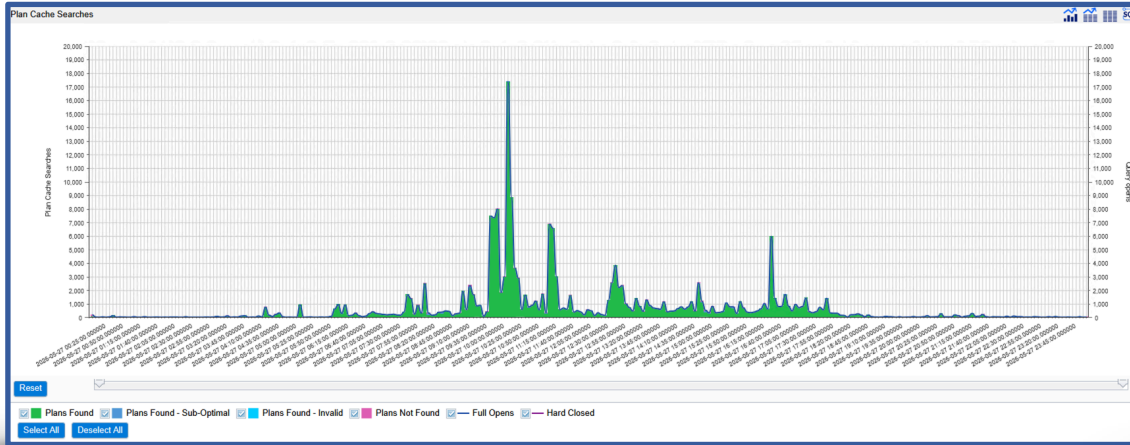
# Plan Cache Snapshots

Plan Cache Snapshots Navigator for i - will be back soon 😊



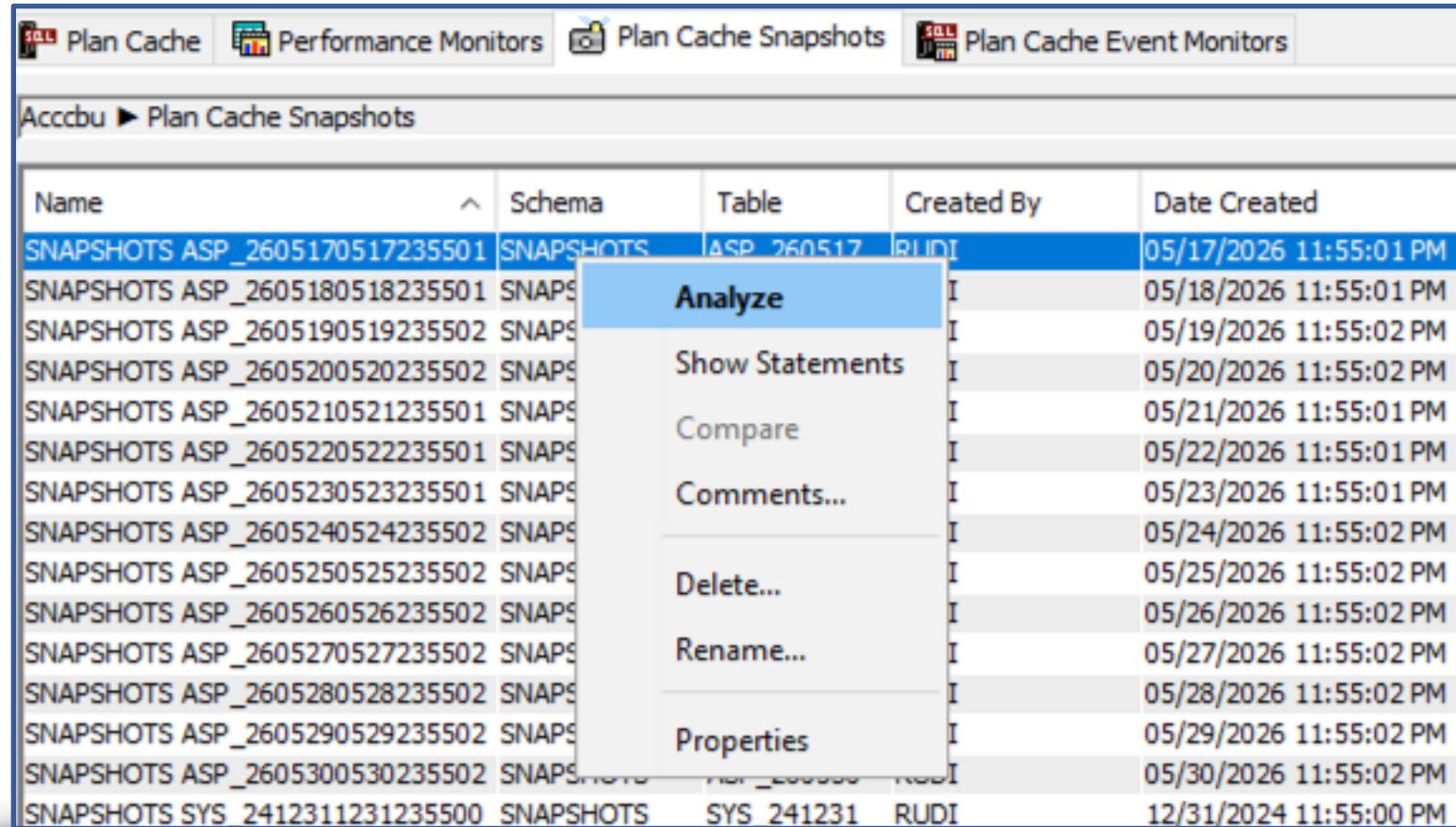
# Plan Cache Perspectives - Currently available

Package Name ↑↓	Perspective Path ↑↓	Perspective ↑↓	Description ↑↓
4 items selected	32 items selected	plan	Filter
Database	SQL Performance Data > Collection Services	<a href="#">Plan Cache Searches</a>	This chart shows the total number of plan cache searches done over time for the selected collection or collections. The total number of plan cache searches is broken down into searches resulting in plans found and plans not found. The number of query full opens and queries hard closed is also shown.
Database	SQL Performance Data > Collection Services	<a href="#">Plan Cache Hit Ratio</a>	This chart shows the plan cache hit ratio and total number of plans in the SQL plan cache.
Database	SQL Performance Data > Collection Services	<a href="#">Plans Detailed</a>	This chart shows the number of plans stored in the SQL plan cache, plans built, plans pruned due to plan cache size, and plans removed from SQL plan cache over time for the selected collection or collections. This chart also shows the size of the SQL plan cache and threshold value for SQL plan cache. Plan cache size threshold is the maximum size that the SQL plan cache is allowed to be before DB2 automatically manages the SQL plan cache and replaces older plans with new plans.



# Plan Cache Snapshots

## Full Table Scans



Plan Cache Snapshots

Name	Schema	Table	Created By	Date Created
SNAPSHOTS ASP_2605170517235501	SNAPSHOTS	ASP_260517	RUDI	05/17/2026 11:55:01 PM
SNAPSHOTS ASP_2605180518235501	SNAPSHOTS	ASP_260518	RUDI	05/18/2026 11:55:01 PM
SNAPSHOTS ASP_2605190519235502	SNAPSHOTS	ASP_260519	RUDI	05/19/2026 11:55:02 PM
SNAPSHOTS ASP_2605200520235502	SNAPSHOTS	ASP_260520	RUDI	05/20/2026 11:55:02 PM
SNAPSHOTS ASP_2605210521235501	SNAPSHOTS	ASP_260521	RUDI	05/21/2026 11:55:01 PM
SNAPSHOTS ASP_2605220522235501	SNAPSHOTS	ASP_260522	RUDI	05/22/2026 11:55:01 PM
SNAPSHOTS ASP_2605230523235501	SNAPSHOTS	ASP_260523	RUDI	05/23/2026 11:55:01 PM
SNAPSHOTS ASP_2605240524235502	SNAPSHOTS	ASP_260524	RUDI	05/24/2026 11:55:02 PM
SNAPSHOTS ASP_2605250525235502	SNAPSHOTS	ASP_260525	RUDI	05/25/2026 11:55:02 PM
SNAPSHOTS ASP_2605260526235502	SNAPSHOTS	ASP_260526	RUDI	05/26/2026 11:55:02 PM
SNAPSHOTS ASP_2605270527235502	SNAPSHOTS	ASP_260527	RUDI	05/27/2026 11:55:02 PM
SNAPSHOTS ASP_2605280528235502	SNAPSHOTS	ASP_260528	RUDI	05/28/2026 11:55:02 PM
SNAPSHOTS ASP_2605290529235502	SNAPSHOTS	ASP_260529	RUDI	05/29/2026 11:55:02 PM
SNAPSHOTS ASP_2605300530235502	SNAPSHOTS	ASP_260530	RUDI	05/30/2026 11:55:02 PM
SNAPSHOTS SYS_2412311231235500	SNAPSHOTS	SYS_241231	RUDI	12/31/2024 11:55:00 PM

# Plan Cache Snapshots

## Full Table Scans

Metric	Value	Reports
SQL Statements	81,517	< Select a report >
Users	6	< Select a report >
Jobs	29	< Select a report >
Threads	29	
Average Table Rows	845,812.823	
Average Rows Returned	521.111	
Average Runtime	1.281123	
Average Parallel Degree Used	1.00	
Maximum Parallel Degree	1.00	
SQE	81,517	< Select a report >
CQE	0	
System Naming	75	< Select a report >
SQL Naming	8	< Select a report >
Unique Open Statements	50	< Select a report >
Full Opens	1,561	< Select a report >
Pseudo Opens	79,956	< Select a report >
Table Scans	71,533	< Select a report >
Average MQTs Used	0.000	< Select a report >
Average Indexes Used	2.150	Table Scan Summary
Full Indexes Created	0	Table Scan Statements
Sparse Indexes Created	0	
Index From Index Created	0	
Index Creates Advised	4,152	< Select a report >
Advised Statistics	1	< Select a report >
Temporary Tables	68,386	< Select a report >
Sorts	10,951	< Select a report >
Access Plans Rebuilt	100	< Select a report >
Sort Sequence	0	
Call Statements	0	
Error	0	

# Plan Cache Snapshots

## Full Table Scans

SNAPSHOTS ASP\_2605170517235501 - Table Scan Statements - [REDACTED]

File Edit View Actions Help

Start Time	Estimated Processing Time	Reason Code	Rows In Base Table	Actual Table Size	Table Function Cardinality	Estimated Rows Selected
2026-05-01 16:43:02.081790	39	Optimizer chose table scan over indexes	140422205	60410422205	60410422205	140422205
2026-05-04 12:19:31.098465	1	Optimizer chose table scan over indexes	25			
2026-05-07 14:24:56.252102	1	Optimizer chose table scan over indexes				
2026-05-07 14:24:56.252102	1	Optimizer chose table scan over indexes				
2026-05-07 14:24:58.107951	1	Optimizer chose table scan over indexes				
2026-05-01 08:50:05.920246	1	Optimizer chose table scan over indexes				
2026-05-01 08:50:05.920246	1	Optimizer chose table scan over indexes				
2026-05-01 08:50:05.920246	1	Optimizer chose table scan over indexes				
2026-05-01 08:50:05.920246	1	Optimizer chose table scan over indexes				
2026-05-01 08:50:05.920246	1	Optimizer chose table scan over indexes				
2026-05-01 08:50:05.920246	1	Optimizer chose table scan over indexes				
2026-05-01 08:50:05.920246	1	Optimizer chose table scan over indexes				
2026-05-01 08:50:05.920246	1	Optimizer chose table scan over indexes				
2026-05-01 08:50:05.920246	1	Optimizer chose table scan over indexes				
2026-05-01 08:50:05.920246	1	Optimizer chose table scan over indexes				
2026-05-01 08:50:05.920246	1	Optimizer chose table scan over indexes				
2026-05-01 15:13:19.008108	1	Optimizer chose table scan over indexes				

**Visual Explain**

Work with SQL Statement

Work with SQL Statement and Variables

SQL Statements with the Same Join Field

SQL Statements with the Same Statement Text

Save to New

Specific Table/View Statements

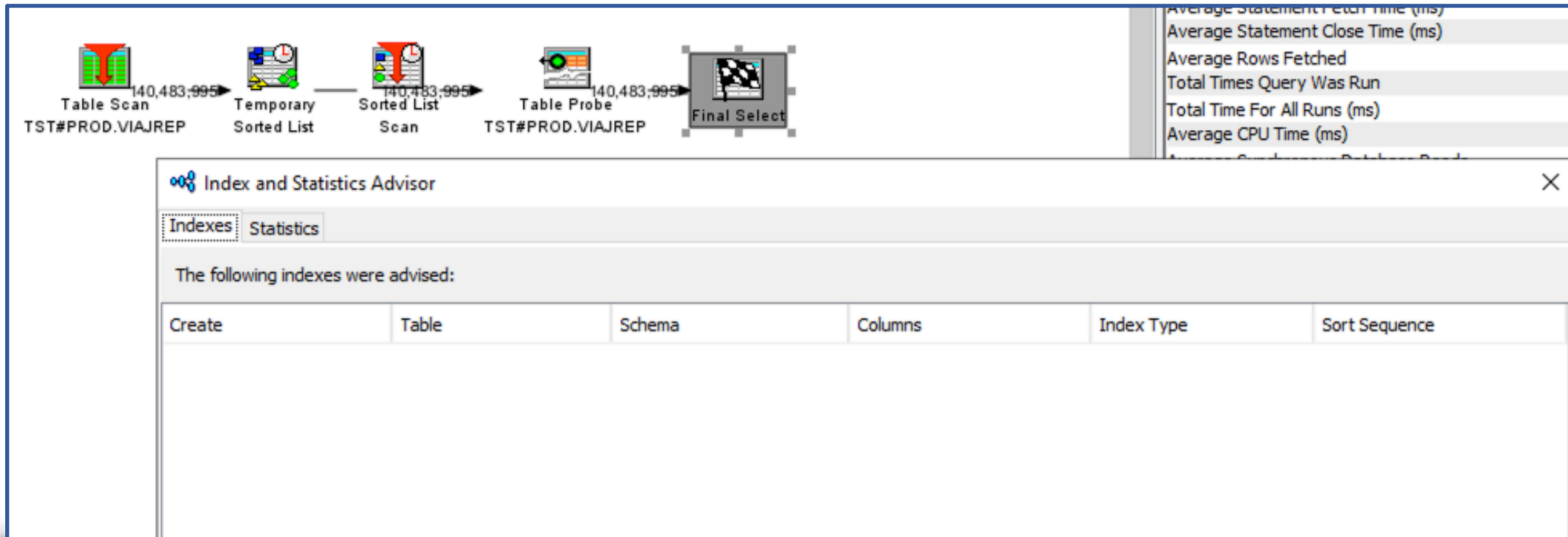
Specific Base Table Statements

---

Table Actions >

# Plan Cache Snapshots

## Full Table Scans



# Plan Cache Snapshots

## Full Table Scans

The screenshot displays a database management tool interface. At the top, a plan cache snapshot is visible, showing a sequence of operations: a 'Table Definition' node, a 'Table Probe' node (labeled 'TST#PROD.VIAJREP' with a value of 140,483,995), and a 'Final Select' node. A context menu is open over the 'Table Definition' node, listing options: 'Table Definition', 'Table Description', 'Statistics Data', 'Show Indexes', 'Show Materialized Query Tables', and 'Show Related'. Below this, a table window shows the metadata for the 'VIAJREP' table. The table has columns for Name, System Name, Owner, Definer, Last Altered, Type, Partitioned, History Version, Text, Size, Row Count, and Deleted Row Count. The 'Deleted Row Count' column is highlighted with a red box, showing a value of 345.

Name	System Name	Owner	Definer	Last Altered	Type	Partitioned	History Version	Text	Size	Row Count	Deleted Row Count
VIAJREP	VIAJREP	VISIONGRP	VISIONGRP	12/13/2017 10:03:49 PM				transaktiehis...	60.417.204.224	140.484.124	345

# Plan Cache Snapshots

## [Database monitor view 3000 - Table Scan](#)

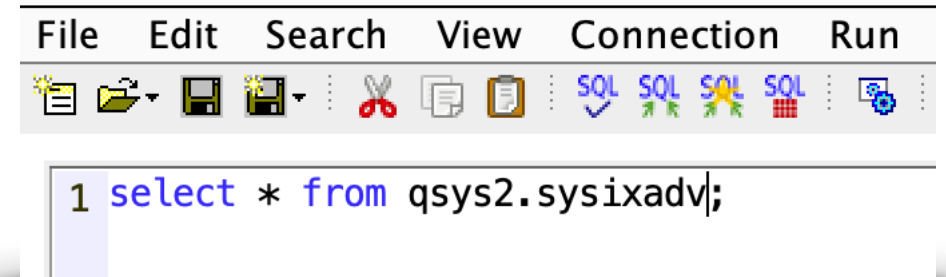
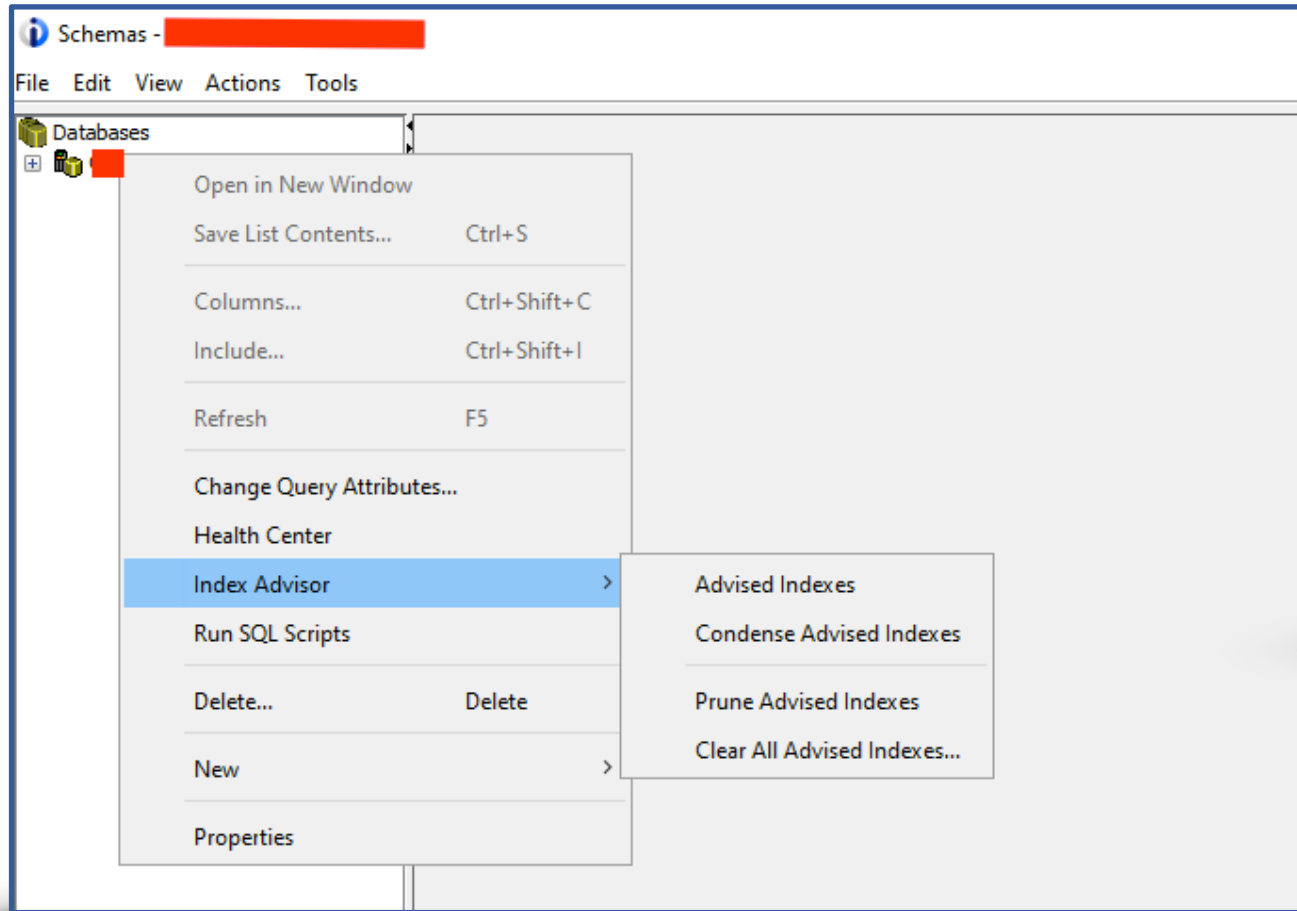
System_Table_Schema	QQTLN	Schema of table queried
System_Table_Name	QQTFN	Name of table queried
Member_Name	QQTMN	Member name of table queried
System_Base_Table_Schema	QQPTLN	Schema name of base table
System_Base_Table_Name	QQPTFN	Name of base table for table queried
Base_Member_Name	QQPTMN	Member name of base table
Table_Total_Rows	QQTOTR	Total rows in table
Estimated_Rows_Selected	QQREST	Estimated number of rows selected
Estimated_Join_Rows	QQAJN	Estimated number of joined rows
Estimated_Processing_Time	QQEPT	Estimated processing time, in seconds

# Plan Cache Snapshots

Two options to avoid the impact of Table Scans on Performance:

1. Use the Index Advisor with or without Visual Explain to avoid Table Scan
2. Reorganize Table to mitigate the impact of a Table Scan
3. Implement Db2 Symmetric Multiprocessing

# Use the Index Advisor with or without Visual Explain



# Use the Index Advisor with or without Visual Explain

Advised Indexes for [REDACTED]

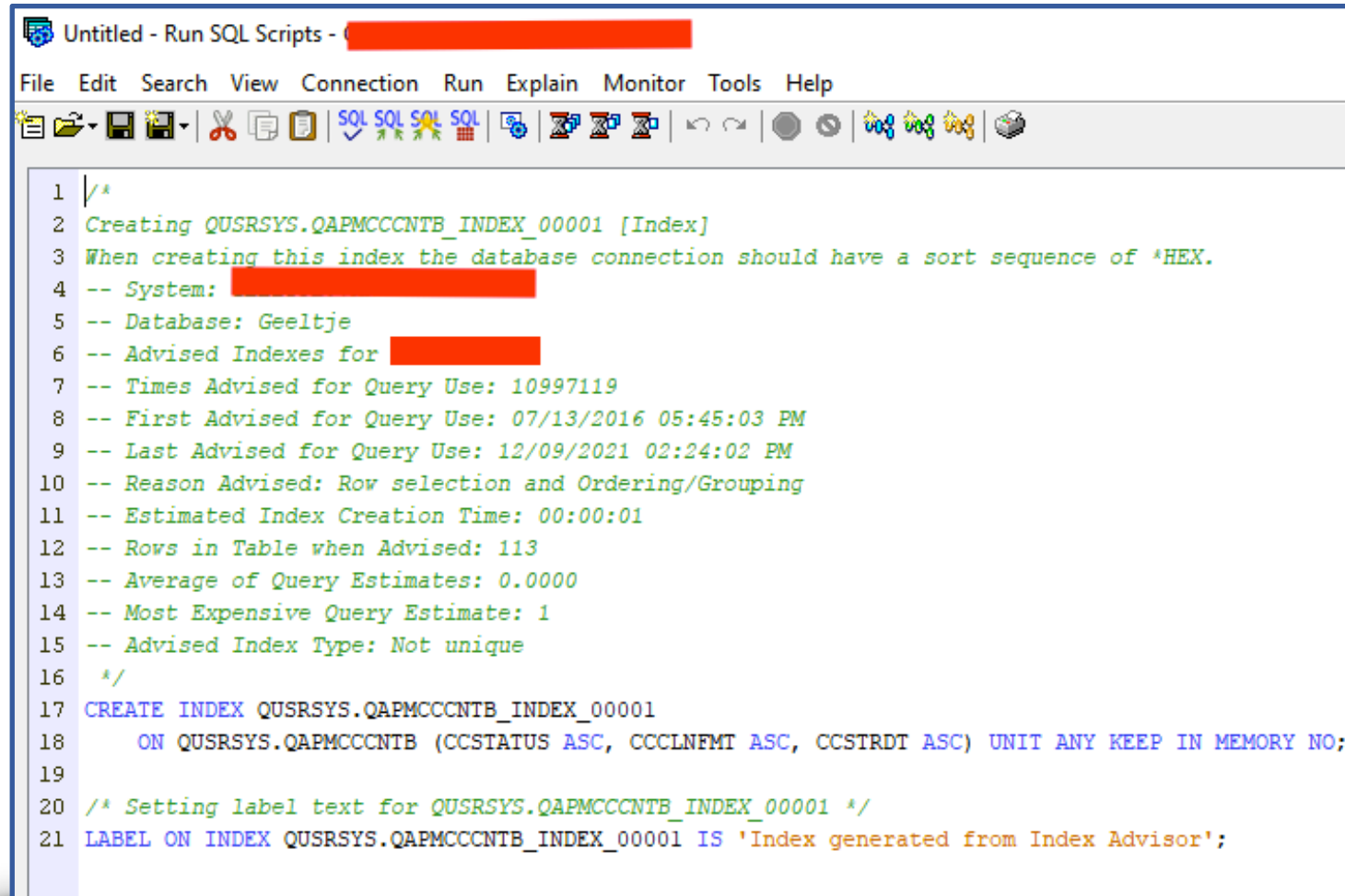
File Edit View Actions

Advised Indexes for [REDACTED]

Table for Which Index was Advised	System Name	Schema	System Schema	Partition	Keys Advised	Leading Keys Order Independent	Advised Index Type	Last Ad Use
QAPMCCCNB	QAPMCCCNB	QUSRSYS	QUSRSYS	For all partitions	CCSTATUS, CCCLNFMT, CCSTRDT	CCSTATUS, CCCLNFMT	Binary Radix	12/09/21 ^
QINAVMNRG	QINAVMNRG	QNAVSRV	QNAVSRV	For all partitions	OBSOLETE	OBSOLETE	Binary Radix	12/09/21
QAPMDISKRB	QAPMDISKRB	QPFRTMP	QPFRTMP	For all partitions	INTNUM, DSDRN	INTNUM, DSDRN	Binary Radix	12/09/21
QAPMDISK	QAPMDISK	QPFRTMP	QPFRTMP	For all partitions	DSIP, DSARM	DSIP, DSARM	Binary Radix	12/09/21
QAPMDISK	QAPMDISK	QPFRTMP	QPFRTMP	For all partitions	INTNUM, DSARM, DATETIME, UTCTIME	INTNUM, DSARM, DATETIME, UTCTIME	Binary Radix	12/09/21
UOA_CACHECO...	UOA_C00001	QWQREPOS	QWQREPOS	For all partitions	INVALIDTIME		Binary Radix	12/09/21
QAPMDISK	QAPMDISK	QPFRTMP	QPFRTMP	For all partitions	INTNUM, DSASP	INTNUM	Binary Radix	12/09/21
QAPMJOBOS	QAPMJOBOS	QPFRTMP	QPFRTMP	For all partitions	INTNUM, DATETIME, UTCTIME, JBPTDE	INTNUM, DATETIME, UTCTIME, JBPTDE	Binary Radix	12/09/21
QAPMJOBMI	QAPMJOBMI	QPFRTMP	QPFRTMP	For all partitions	INTNUM, JBPTDE	INTNUM, JBPTDE	Binary Radix	12/09/21
QAPMJOBMI	QAPMJOBMI	QPFRTMP	QPFRTMP	For all partitions	INTNUM, DATETIME, UTCTIME, JBPTDE	INTNUM, DATETIME, UTCTIME, JBPTDE	Binary Radix	12/09/21
QAPMJOBMI	QAPMJOBMI	QPFRTMP	QPFRTMP	For all partitions	INTNUM, JBPTDE		Binary Radix	12/09/21
QAPMHDDISK	QAPMHDDISK	QPFRHIST	QPFRHIST	For all partitions	DATETIME		Binary Radix	12/09/21
QWQAACRTE	QWQAACRTE	QWQREPOS	QWQREPOS	For all partitions	WQRTEUSR	WQRTEUSR	Binary Radix	12/09/21
QWQARTMENV	QWQARTMENV	QWQREPOS	QWQREPOS	For all partitions	WQRTEID	WQRTEID	Binary Radix	12/09/21
QAPMJOBOS	QAPMJOBOS	QPFRTMP	QPFRTMP	For all partitions	JBTYPE, INTNUM, JBPTDE	JBTYPE, INTNUM	Binary Radix	12/09/21
QAPMDISK	QAPMDISK	QPFRTMP	QPFRTMP	For all partitions	DSASP		Binary Radix	12/09/21
QAEZD0001D	QAEZD0001D	QRPLOBJ	QRPLOBJ	For all partitions	QEZDIRFSID, QEZDIRFID	QEZDIRFSID, QEZDIRFID	Binary Radix	11/08/21
QAPMJOBMI	QAPMJOBMI	QPFRTMP	QPFRTMP	For all partitions	JBTCPU		Binary Radix	12/09/21
QAPMJOBMI	QAPMJOBMI	QPFRTMP	QPFRTMP	For all partitions	INTNUM, JBPTDE	INTNUM	Binary Radix	12/09/21
QAPMHDJOBM	QAPMHDJOBM	QPFRHIST	QPFRHIST	For all partitions	DATETIME		Binary Radix	12/09/21
QAPMJOBMI	QAPMJOBMI	QPFRTMP	QPFRTMP	For all partitions	JBTYPE, INTNUM, JBTCPU		Binary Radix	12/09/21
QAPMJOBOS	QAPMJOBOS	QPFRTMP	QPFRTMP	For all partitions	JBTYPE, INTNUM, JBINTR	JBTYPE, INTNUM	Binary Radix	12/09/21
QAPMJOBMI	QAPMJOBMI	QPFRTMP	QPFRTMP	For all partitions	JBTYPE, INTNUM, JBPTDE	JBTYPE, INTNUM	Binary Radix	12/09/21
QAPMJOBMI	QAPMJOBMI	QPFRTMP	QPFRTMP	For all partitions	JBTYPE, INTNUM, JBTCPU	JBTYPE, INTNUM	Binary Radix	12/09/21 v

100 rows retrieved (more data available).

# Use the Index Advisor with or without Visual Explain



```
1 /*
2 Creating QUSRSYS.QAPMCCNTB_INDEX_00001 [Index]
3 When creating this index the database connection should have a sort sequence of *HEX.
4 -- System: ██████████
5 -- Database: Geeltje
6 -- Advised Indexes for ██████████
7 -- Times Advised for Query Use: 10997119
8 -- First Advised for Query Use: 07/13/2016 05:45:03 PM
9 -- Last Advised for Query Use: 12/09/2021 02:24:02 PM
10 -- Reason Advised: Row selection and Ordering/Grouping
11 -- Estimated Index Creation Time: 00:00:01
12 -- Rows in Table when Advised: 113
13 -- Average of Query Estimates: 0.0000
14 -- Most Expensive Query Estimate: 1
15 -- Advised Index Type: Not unique
16 */
17 CREATE INDEX QUSRSYS.QAPMCCNTB_INDEX_00001
18     ON QUSRSYS.QAPMCCNTB (CCSTATUS ASC, CCCLNFMT ASC, CCSTRDT ASC) UNIT ANY KEEP IN MEMORY NO;
19
20 /* Setting label text for QUSRSYS.QAPMCCNTB_INDEX_00001 */
21 LABEL ON INDEX QUSRSYS.QAPMCCNTB_INDEX_00001 IS 'Index generated from Index Advisor';
```

# Use the Index Advisor with or without Visual Explain

Advised Indexes for XXXXXXXXXX

File Edit View Actions

Advised Indexes for Geeltje

Most Expensive Query Estimate	Average of Query Estimates	Rows in Table when Advised	NLSS Table Advised	NLSS Schema Advised	MTI Used	MTI Created	MTI Last Used	MTI Used for Statistics	MTI Last Used for Statistics	EVI Distinct Values	Firs Qu
64	1	0.0000	113 *HEX		10301138	31 12/09/2021 02:25:09 PM		0			07/
64	1	0.0000	229 *HEX		5141987	36 12/09/2021 02:25:09 PM		1	12/09/2021 02:25:09 PM		05/
64	1	0.0000	192 *HEX		149454	181 12/01/2021 00:02:07 AM		0			06/
64	1	0.0000	192 *HEX		149503	182 12/01/2021 00:02:07 AM		0			06/
64	1	0.0000	192 *HEX		0	0		0			06/
64	1	0.0000	0 *HEX		743618	8 12/09/2021 01:02:00 AM		0			12/
64	1	0.0000	192 *HEX		144	0 12/01/2021 00:02:07 AM		0			06/
64	1	0.0000	9608 *HEX		0	0		0			06/
64	1	0.0000	48003 *HEX		553003	553 12/09/2021 00:17:04 AM		1	12/09/2021 00:16:56 AM		06/
64	1	0.0000	48003 *HEX		77087	327 12/09/2021 00:15:09 AM		0			06/
64	1	0.0009	48003 *HEX		0	0		1	12/09/2021 00:17:01 AM		06/
64	234	0.0001	13032 *HEX		423765	25 12/09/2021 00:17:05 AM		1	12/09/2021 08:00:02 AM		07/
64	1	0.0000	4 *HEX		0	0		0			11/
64	1	0.0000	1 *HEX		2	1 01/15/2019 04:07:03 PM		0			11/
64	1	0.0000	9608 *HEX		144	1 11/27/2019 00:03:54 AM		0			06/
64	1	0.0000	192 *HEX		35552	0 12/01/2021 00:02:07 AM		0			06/
64	1	0.0000	58717 *HEX		0	0		0			11/
64	1	0.0000	48003 *HEX		263280	641 12/09/2021 00:17:04 AM		1	12/09/2021 00:17:00 AM		06/
64	1	0.0000	48003 *HEX		328857	192 12/09/2021 00:17:04 AM		1	12/09/2021 00:16:57 AM		06/
64	792	0.0200	101081 *HEX		369945	16 12/09/2021 00:17:06 AM		1	12/09/2021 00:17:05 AM		07/
64	1	0.0010	48003 *HEX		0	0		1	12/09/2021 00:17:01 AM		06/
64	1	0.0000	9608 *HEX		120562	60 11/30/2021 00:03:13 AM		0			06/
64	1	0.0000	48003 *HEX		772	8 05/07/2020 00:12:11 AM		0			06/
64	1	0.0000	48003 *HEX		217597	485 12/09/2021 00:16:56 AM		0			06/

100 rows retrieved (more data available).

# Use the Index Advisor with or without Visual Explain

## [Maintained Temporary Index \(MTI\)](#)

On this page

- Maintained Temporary Indexes briefly explained**
- Monitoring for temporary storage
- Monitoring for optimal query performance
- Drilling down for details
- Index Advisor
- MTI\_INFO and the SQL Performance Center
- Managing MTIs
- Solve a scenario
- Maintenance-free MTIs
- Related topics

## Maintained Temporary Indexes briefly explained

In [part 1 of this series](#), we described the kinds of temporary storage that the SQL optimizer uses. We showed how the QSYS2.SYSTMPSTG system view provides a high-level overview of temporary storage across the system. There are five *global buckets* that directly reflect usage by the SQL optimizer. Of these buckets, this article focuses on the one labeled, **\*DATABASE DSI SQE MTI**. This bucket reports the amount of storage used across the system by maintained temporary indexes.

[Maintained Temporary Indexes](#) (MTIs) are indexes that the query optimizer creates without any user intervention. They function like any other index on the system. Internally, they are radix indexes, just like permanent (that is, user created) indexes, and they provide the ability to read rows in keyed order or to probe for specific values. Like a permanent index, the actual size of an MTI depends on the keys included in the index and on the underlying data. They are actively maintained, meaning that any change in the dataspace is immediately reflected in the MTI. In many cases, MTIs can be shared between queries, just like permanent indexes. But they differ from permanent indexes in two important ways: they use temporary storage, and they are managed entirely by the optimizer. They are created whenever the optimizer needs them and deleted when the optimizer is finished with them.

In simplest terms, there is only one reason for the optimizer to build an MTI: no suitable permanent index exists to meet the requirements of a query. But, if we drill down into this single basic reason, we'll find a couple of distinct circumstances that cause the optimizer to need an index. Understanding each circumstance—and whether it applies to your workload—is crucial to effectively managing your usage of MTIs.

Time to read: 17 minutes

# Use the Index Advisor with or without Visual Explain

The screenshot shows the IBM DB2 Index Advisor interface. The 'Maintained Temporary Indexes' window is open, displaying a table of index statistics. A red arrow points to the 'Maintained Temporary Indexes' tab in the top navigation bar.

Name	Table Name	Table Partition	Table System Schema	Reference Count	Number of Keys	Key Definition	State
MTI(160)(249591)	V5M#REP	V5M#REP	VIS#PROD	12	2	MHECS1, MHLB0T	Valid
MTI(292)(993466)	V6OFREP	V6OFREP	VIS@DWH	16	3	OFZONC, OFZZNC, OFZYNC	Valid
MTI(85)(38087)	WAEGREP	WAEGREP	VIS#PROD	20	3	FCI_VERWERKT, FCI_TYPE, FCI_CONCERN	Valid
MTI(95)(16374)	VICFREP	VICFREP	VIS#PROD	189	2	CFCKOD, CFWHNB	Valid
MTI(52)(46341)	VICZREP	VICZREP	VIS#PROD	2	2	CZBQTX, CZDKDT	Valid
MTI(29)(9523)	VNEBREP	VNEBREP	VIS#PROD	6	4	EBN3NC, EBNSKD, EBOUDT, EBN#NR	Valid
MTI(327)(750698)	QAEZD00033	QCURRENT	QUSRSYS	2	1	DIOBLI	Valid
MTI(19)(63489)	V5MTREP	V5MTREP	VIS#ALG	30	1	MTAATX	Valid
MTI(78)(37474)	V6OVREP	V6OVREP	VIS#PROD	71	1	OVHIVF	Valid
MTI(33)(16361)	V9AVREP	V9AVREP	VIS#ALG	47	3	BDR_BDRNR_, CFA_ORDERNUMMER_KLANT, CFA_DEPOTDATUM	Valid
MTI(154)(264856)	VICJREP	VICJREP	VIS#PROD	253	1	CJHSST	Valid
MTI(10)(63704)	V7R4REP	V7R4REP	VIS#ALG	8	1	R4BSTX	Valid
MTI(41)(654514)	V4JRREP	V4JRREP	VIS#KLOK	2	1	JRDJCE	Valid
MTI(159)(942994)	V4KDREP	V4KDREP	VIS#PROD	6	2	KDBOTY, KDPJNO	Valid
MTI(37)(654513)	V4JRREP	V4JRREP	VIS#KLOK	3	1	JRALCE	Valid
MTI(36)(874572)	V5MSREP	V5MSREP	VIS#ALG	33	2	MSIMNB, MSPQCD	Valid
MTI(37)(22272)	V9AVREP	V9AVREP	VIS#ALG	8	1	CFA_FACTUURNUMMER	Valid
MTI(36)(511846)	V5MSREP	V5MSREP	VIS#ALG	4	1	MSAATX	Valid
MTI(23)(63490)	V5MSREP	V5MSREP	VIS#ALG	53	1	MSIMNB	Valid
MTI(171)(942995)	V4KDREP	V4KDREP	VIS#PROD	9	1	KDPJNO	Valid
MTI(26)(654571)	V6N9REP	V6N9REP	VIS#KLOK	2	1	N9E7VF	Valid
MTI(87)(905211)	V7QFREP	V7QFREP	VIS@DWH	5	1	QFJ4ND	Valid
MTI(29)(890126)	VNEBREP	VNEBREP	VIS#VDF	8	4	EBN3NC, EBNSKD, EBOUDT, EBN#NR	Valid
MTI(112)(11444)	VIBIREP4	VIBIREP4	VIS#KLOK	8	2	B1GZST, B1B6KD	Valid
MTI(46)(654671)	VIAAREP	VIAAREP	VIS#KLOK	2	1	AAFGTX	Valid
MTI(8)(106643)	V9A3REP	V9A3REP	VIS#KLOK	2	1	A3GK52	Valid
MTI(113)(55929)	VIBIREP4	VIBIREP4	VIS#PROD	9	2	B1GZST, B1B6KD	Valid
MTI(29)(778645)	VNEBREP	VNEBREP	VIS#KLOK	6	4	EBN3NC, EBNSKD, EBOUDT, EBN#NR	Valid
MTI(48)(553853)	VIBJREP	VIBJREP	VIS#PROD	14	1	B1DZST	Valid
MTI(104)(8342)	VIAAREP	VIAAREP	VIS#KLOK	20	1	AAWB0D	Valid
MTI(21)(525571)	V7R2REP	V7R2REP	VIS#ALG	10	3	R2IMNB, R2PQCD, R2PQDT	Valid
MTI(42)(654670)	VIAAREP	VIAAREP	VIS#KLOK	2	1	AAFMCD	Valid
MTI(85)(55924)	VIBJREP	VIBJREP	VIS#PROD	371	2	B1DYST, B1JWNB	Valid
MTI(81)(63756)	VICHREP	VICHREP	VIS#PROD	53	2	CHW0CD, CHAZTX	Valid

100 rows retrieved (more data available).

# Use the Index Advisor with or without Visual Explain

Maintained Temporary Indexes

Name	Table Name	Table Partition	Table System Schema	Reference Count	Number of Keys	Key Definition	State
MTI(160)(249591)	V5MIHREP	V5MIHREP	VIS#PROD	12	2	MHECS1, MHLBDT	Valid
MTI(292)(99)	Create Index	REP	VIS@DWH	16	3	OFZ0NC, OFZZNC, OFZYNC	Valid
MTI(85)(380)		REP	VIS#PROD	20	3	FCI_VERWERKT, FCI_TYPE, FCI_CONCERN	Valid
MTI(95)(163)		EP	VIS#PROD	189	2	CFCCCKD, CFWHNB	Valid
MTI(52)(463)		EP	VIS#PROD	2	2	CZBQTX, CZDKDT	Valid
MTI(29)(952)	Table	REP	VIS#PROD	6	4	EBN3NC, EBN5KD, EBDUDT, EBN4NR	Valid
MTI(327)(75)		RENT	QUSRSYS	2	1	DIOBLI	Valid
MTI(19)(63489)	V5MTREP	V5MTREP	VIS#ALG	30	1	MTAATX	Valid
MTI(78)(37474)	V6OVREP	V6OVREP	VIS#PROD	71	1	OVHIVF	Valid
MTI(33)(16361)	V9AVREP	V9AVREP	VIS#ALG	47	3	BDR_BDRNR_, CFA_ORDERNUMMER_KLANT, CFA_DEPOTDATUM	Valid
MTI(154)(264856)	VICJREP	VICJREP	VIS#PROD	253	1	CJH5ST	Valid
MTI(10)(63704)	V7R4REP	V7R4REP	VIS#ALG	8	1	R4B5TX	Valid
MTI(41)(654514)	V4JRREP	V4JRREP	VIS#KLOK	2	1	JRDICE	Valid
MTI(159)(942994)	V4KDREP	V4KDREP	VIS#PROD	6	2	KDBOTY, KDPJNO	Valid
MTI(37)(654513)	V4JRREP	V4JRREP	VIS#KLOK	3	1	JRALCE	Valid
MTI(36)(874572)	V5MSREP	V5MSREP	VIS#ALG	33	2	MSIMNB, MSPQCD	Valid
MTI(37)(22272)	V9AVREP	V9AVREP	VIS#ALG	8	1	CFA_FACTUURNUMMER	Valid
MTI(36)(511846)	V5MSREP	V5MSREP	VIS#ALG	4	1	MSAATX	Valid
MTI(23)(63490)	V5MSREP	V5MSREP	VIS#ALG	53	1	MSIMNB	Valid
MTI(171)(942995)	V4KDREP	V4KDREP	VIS#PROD	9	1	KDPJNO	Valid
MTI(26)(654571)	V6N9REP	V6N9REP	VIS#KLOK	2	1	N9E7VF	Valid
MTI(87)(905211)	V7QFREP	V7QFREP	VIS@DWH	5	1	QFJ4ND	Valid
MTI(29)(890126)	VNEBREP	VNEBREP	VIS#VDF	8	4	EBN3NC, EBN5KD, EBDUDT, EBN4NR	Valid
MTI(112)(11444)	VIB1REPC	VIB1REP4	VIS#KLOK	8	2	B1GZST, B1B6KD	Valid
MTI(46)(654671)	VIAAREP	VIAAREP	VIS#KLOK	2	1	AAF6TX	Valid
MTI(8)(106643)	V9A3REP	V9A3REP	VIS#KLOK	2	1	A3GXS2	Valid
MTI(113)(55929)	VIB1REPC	VIB1REP4	VIS#PROD	9	2	B1GZST, B1B6KD	Valid
MTI(29)(778645)	VNEBREP	VNEBREP	VIS#KLOK	6	4	EBN3NC, EBN5KD, EBDUDT, EBN4NR	Valid
MTI(48)(553853)	VIBJREP	VIBJREP	VIS#PROD	14	1	BJDZST	Valid
MTI(104)(8342)	VIAAREP	VIAAREP	VIS#KLOK	20	1	AAWBCD	Valid
MTI(21)(525571)	V7R2REP	V7R2REP	VIS#ALG	10	3	R2ZIMNB, R2PQCD, R2PQDT	Valid
MTI(42)(654670)	VIAAREP	VIAAREP	VIS#KLOK	2	1	AAFMC	Valid
MTI(85)(55924)	VIBJREP	VIBJREP	VIS#PROD	371	2	BJDYST, BJIWNB	Valid
MTI(81)(63756)	VICHREP	VICHREP	VIS#PROD	53	2	CHWXCD, CHAZTX	Valid
MTI(10)(653054)	VICJREP	VICJREP	VIS#PROD	2	1	CJH5ST	Valid

100 rows retrieved (more data available).

## Filters applied:

- Statements identified by QRO hash value: F0F1B51F
- Include statements initiated by the operating system

Laatste Keer Uitgevoerd	Meest Kostbare Tijd (sec)	Totale Verwerking Tijd (sec)	Totaal Aantal keren Uitgevoerd	Gemiddeld Verwerking Tijd (sec)	Instructie
2026-05-30 08:50:00.427589	0.0106	0.0652	12	0.0054	QUERY/400 VIS
2026-05-16 08:50:00.009966	0.0076	0.0384	8	0.0048	QUERY/400 VIS
2026-05-31 08:50:00.710469	0.0022	0.0053	4	0.0013	QUERY/400 VIS
2026-05-18 08:50:00.935349	0.0040	0.0050	3	0.0016	QUERY/400 VIS

# Use the Index Advisor with or without Visual Explain

The screenshot displays the IBM Data Studio interface. The main window shows a query: `select * from qiws.qcustcdt where state='TX'`. A red arrow points from the `state='TX'` condition to the Visual Explain window. The Visual Explain window shows a graph with a red arrow pointing to a table icon labeled "Table Scan QIWS.QCUSTCDT". A second arrow labeled "2" points from the table scan to a "Final" node. The Index and Statistics Advisor window is open, showing the following table:

Create	Table	Schema	Columns	Index Type	Sort Sequence
<input checked="" type="checkbox"/>	QCUSTCDT	QIWS	STATE	BINARY RADIX	None (Sort by hexadecimal value)

The Index and Statistics Advisor window also includes a "Time Information" table:

Attribute	Value
Query Engine Used	SQE
<b>Time Information</b>	
Timestamp for Creation of Monitor Entry	2026-05-31-14.57.48.872445
Statement Start Timestamp	2026-05-31-14.57.48.855248
Statement End Timestamp	2026-05-31-14.57.48.872445
Total Estimated Run Time (ms)	,001

The Index and Statistics Advisor window also has a "Show SQL" button and a "Create..." button. The main window shows the query execution plan and the Index and Statistics Advisor window.

# Use the Index Advisor with or without Visual Explain

The screenshot displays the Visual Explain interface for a query. The execution plan shows the following steps:

- Table Scan PRODDTA.F4201**: Estimated rows: 1,027,329
- Temporary Hash Table**: Receives data from the first table scan.
- Hash Probe**: Receives data from the Temporary Hash Table.
- Table Scan PRODDTA.F4211**: Estimated rows: 1,420,607
- Nested Loop Join**: Receives data from both the Hash Probe and the second table scan. Estimated rows: 1,420,607.
- Final Select**: Receives data from the Nested Loop Join.

Runtime statistics on the right side of the window:

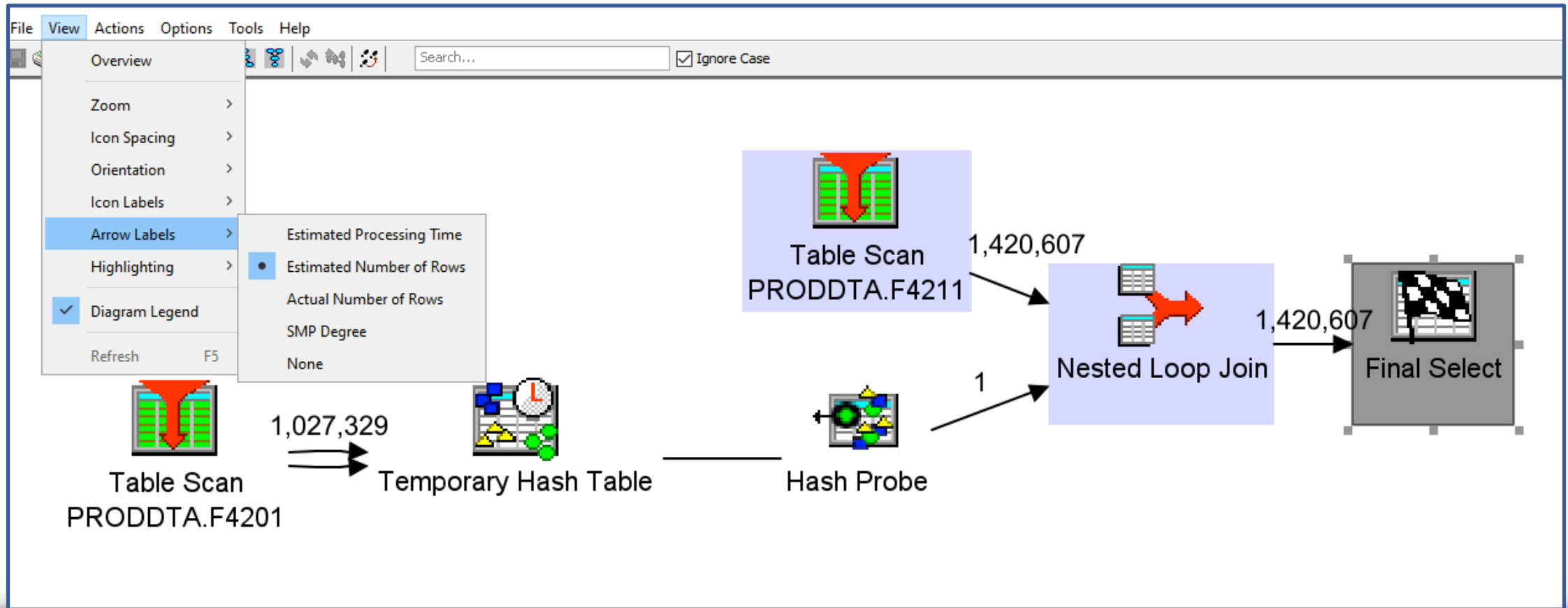
Attribute	Value
Query Engine Used	SQE
<b>Time Information</b>	
Timestamp for Creation of Monitor Entry	2021-11-11-05.25.20.213
Monitor Entry	802
Statement Start Timestamp	2021-11-11-05.25.20.213
Timestamp	802
Statement End Timestamp	2021-11-11-05.25.20.213
Timestamp	802
Total Estimated Run Time (ms)	16,981
<b>Actual Runtime Inform</b>	
Optimization Time (ms)	260
Longest Key Range Estimate (ms)	0
Key Range Estimate Timed Out	No
Run Time (ms)	44,254
Statement Open Time (ms)	Not Available
Statement Fetch Time (ms)	44,254
Statement Close Time (ms)	Not Available
Rows Fetched	1,504,911
Total Times Query Was Run	231
Total Time For All Runs (ms)	1.022E7
CPU Time	9
Synchronous Database	5,649

The SQL statement being executed is:

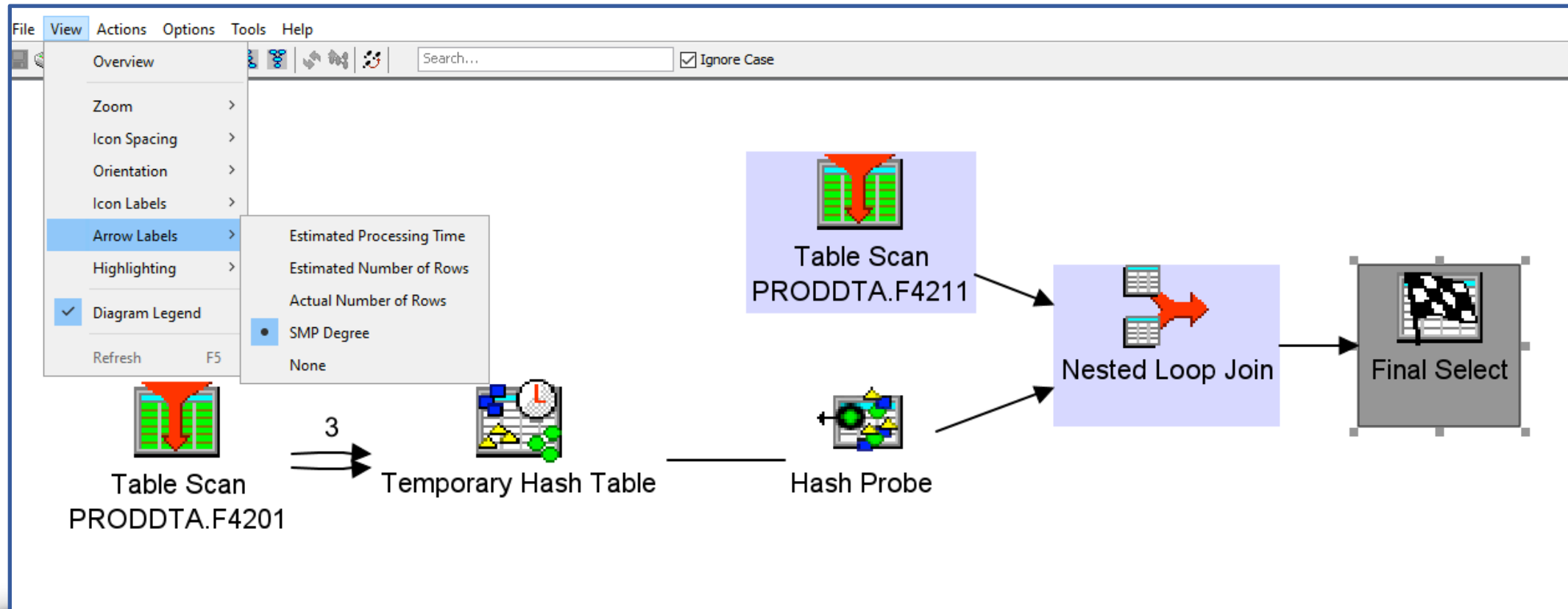
```

SELECT *
FROM PRODDTA.F4211,
PRODDTA.F4201
WHERE SDKCOO = SHRCCO
AND SDDOCO = SHDOCO
AND SDDCTO = SHDCTO
AND SDSFXO = SHSFXO
AND SDLITR < ?
AND (SDDGL = ?
OR SDDGL > ?)
AND SDDCTO < ?
AND SDDCTO < ?
AND SDDCTO < ?
  
```

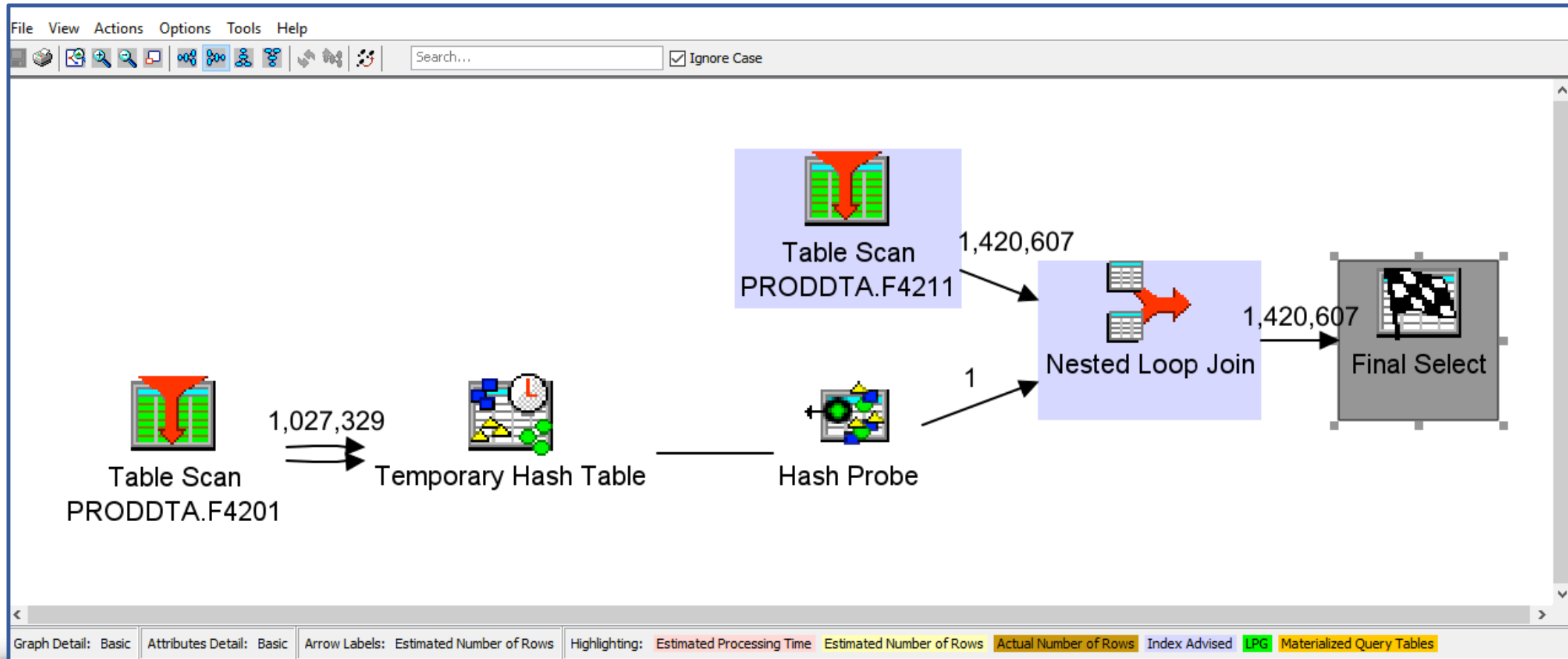
# Use the Index Advisor with or without Visual Explain



# Use the Index Advisor with or without Visual Explain



# Use the Index Advisor with or without Visual Explain



# Use the Index Advisor with or without Visual Explain

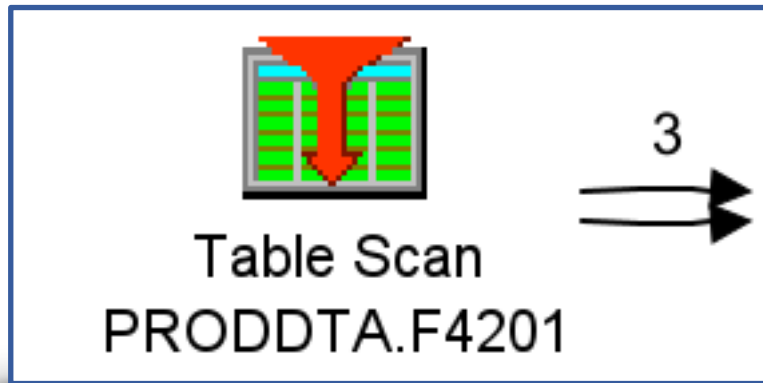
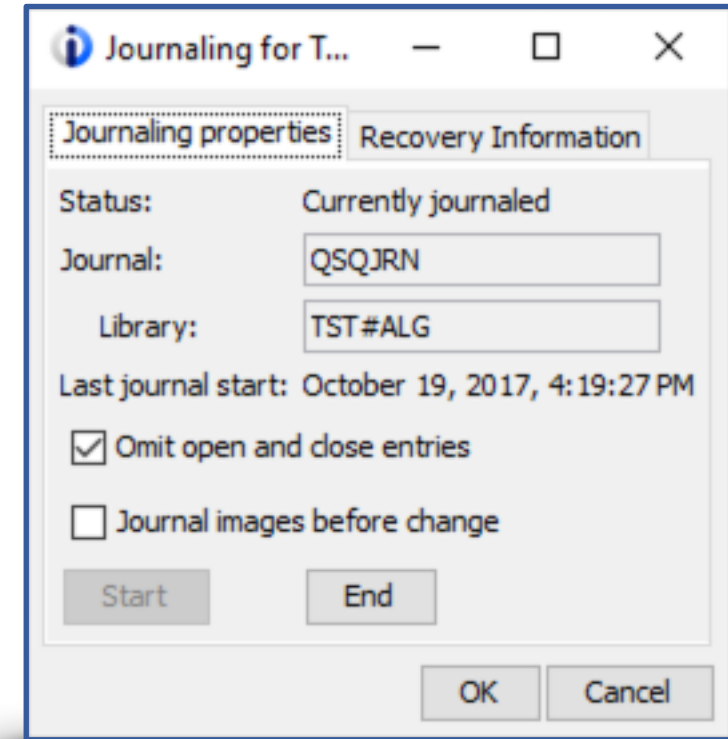
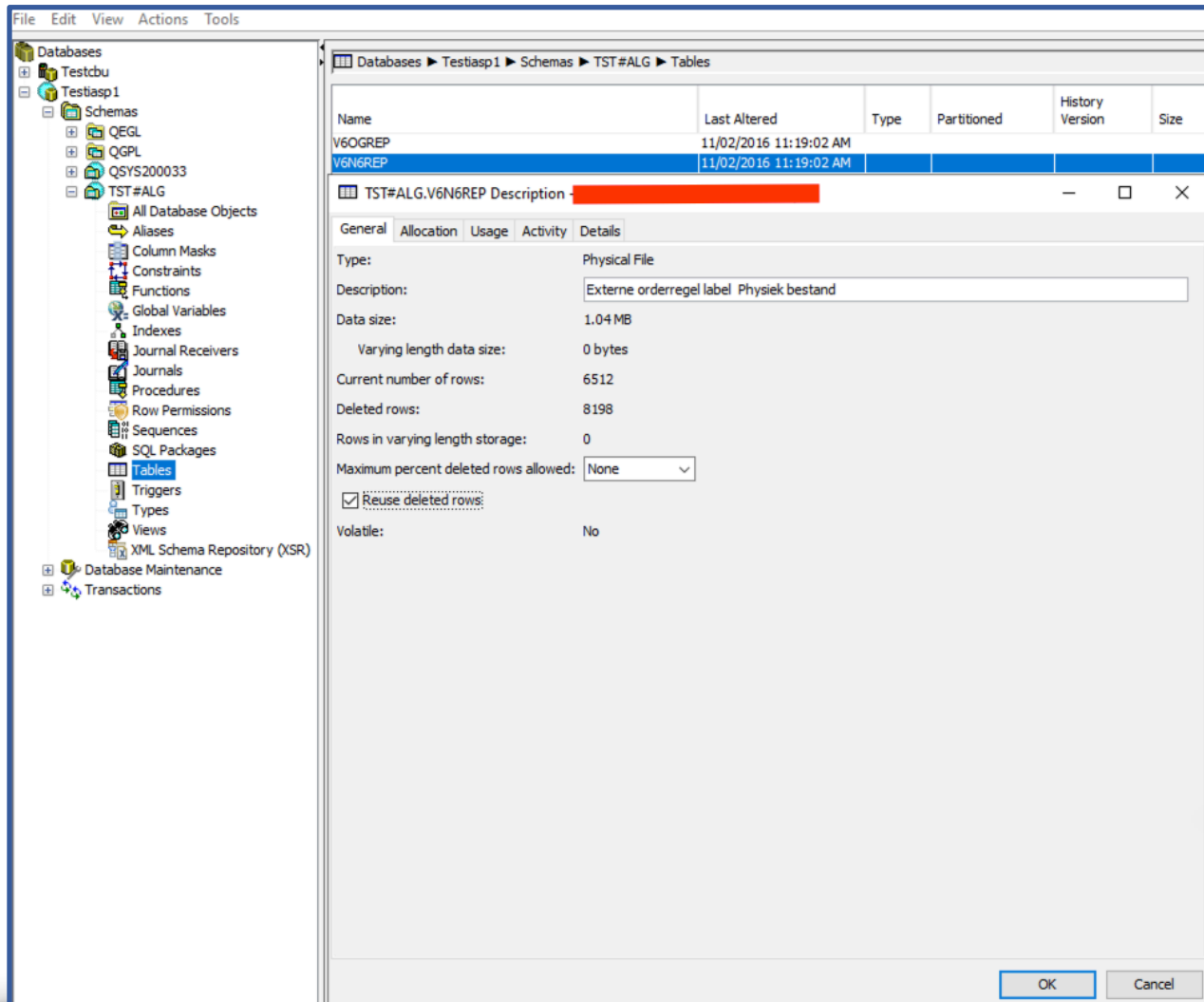


Table 2. Table scan attributes	
<b>Data access method</b>	<b>Table scan</b>
<b>Description</b>	Reads all the rows from the table and applies the selection criteria to each of the rows within the table. The rows in the table are processed in no guaranteed order, but typically they are processed sequentially.
<b>Advantages</b>	<ul style="list-style-type: none"> <li>Minimizes page I/O operations through asynchronous pre-fetching of the rows since the pages are scanned sequentially</li> <li>Requests a larger I/O to fetch the data efficiently</li> </ul>
<b>Considerations</b>	<ul style="list-style-type: none"> <li>All rows in the table are examined regardless of the selectivity of the query</li> <li>Rows marked as deleted are still paged into memory even though none are selected. You can reorganize the table to remove deleted rows.</li> </ul>
<b>Likely to be used</b>	<ul style="list-style-type: none"> <li>When expecting many rows returned from the table</li> <li>When the number of large I/Os needed to scan is fewer than the number of small I/Os required to probe the table</li> </ul>
<b>Example SQL statement</b>	<pre>SELECT * FROM Employee WHERE WorkDept BETWEEN 'A01' AND 'E01' OPTIMIZE FOR ALL ROWS</pre>
<b>Database Monitor and Plan Cache record indicating use</b>	QQRID 3000 - Table Scan
<b>SMP parallel enabled</b>	Yes
<b>Also referred to as</b>	Table Scan, Preload
<b>Visual Explain icon</b>	

[IBM i 7.6 Database Performance and Query Optimization](#)

# Reorganize Table



# Reorganize Table

Reorganize TST#ALG.V6N6REP - [redacted]

Reorganize the table

By compressing out deleted rows

Preserve arrival row sequence

By selected index

Table partition: First partition

Reorganize starting at: First row

Allow reorganization to be suspended

Allow users to access the table during reorganization (Online)

Allow changes to the table during the reorganization

Access paths: Maintain during

Number of tasks: Use current setting

Use current setting

No parallel processing

Show Command OK Cancel

# Reorganize Table

```

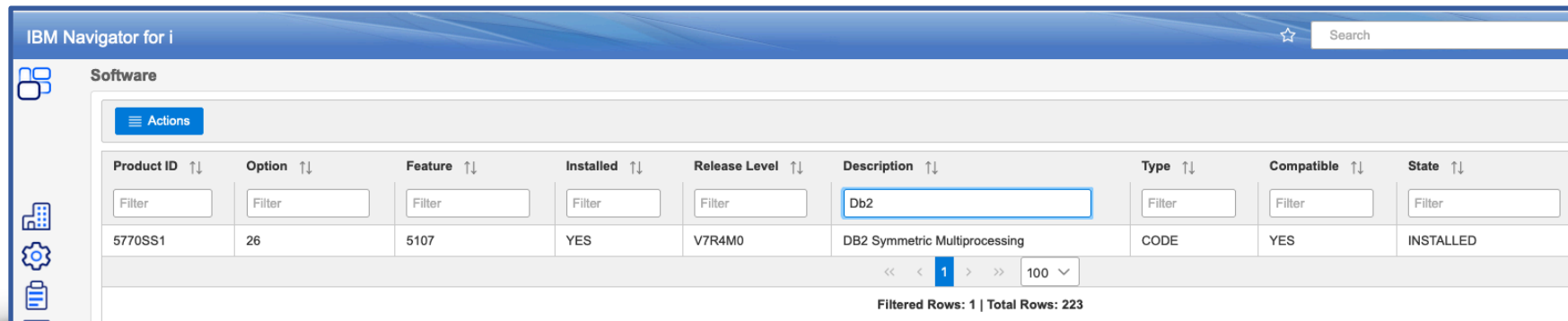
1  -- category: System Management
2  -- description: Reorganize on-the-fly per Schema
3  SELECT SYSTEM_TABLE_SCHEMA,
4         SYSTEM_TABLE_NAME,
5         SYSTEM_TABLE_MEMBER,
6         TEXT_DESCRIPTION,
7         TABLE_PARTITION,
8         PARTITION_TYPE,
9         NUMBER_ROWS,
10        NUMBER_PAGES,
11        OVERFLOW,
12        AVGROWSIZE,
13        NUMBER_DELETED_ROWS,
14        -- 'CL:RGZPFM FILE(' CONCAT TRIM(SYSTEM_TABLE_SCHEMA) CONCAT '/' CONCAT TRIM(SYSTEM_TABLE_NAME) CONCAT ' )
15        MBR(*FIRST) KEYFILE(*RPLDLTRCD) ALWCANCEL(*YES) LOCK(*SHRUPD) RBDACCPH(*NO) FROMRCD(*START)',
16        QSYS2.QCMDXEX('RGZPFM FILE(' CONCAT TRIM(SYSTEM_TABLE_SCHEMA) CONCAT '/' CONCAT TRIM(SYSTEM_TABLE_NAME) CONCAT ' )
17        MBR(*FIRST) KEYFILE(*RPLDLTRCD) ALWCANCEL(*YES) LOCK(*SHRUPD) RBDACCPH(*NO) FROMRCD(*START)') as true,
18        DATA_SIZE
19  FROM QSYS2.SYSMEMBERSTAT
20  WHERE TABLE_SCHEMA = :TABLE_SCHEMA
21         AND NUMBER_DELETED_ROWS <> 0
22         AND NUMBER_DELETED_ROWS >= 100
23  ORDER BY NUMBER_DELETED_ROWS DESC fetch first 40 rows only;
24

```

# Db2 Symmetric Multiprocessing

## [Objects processed in parallel](#)

Database parallelism,  
while inherently part of Db2 for i,  
is enabled by installing the optional IBM i feature  
"Db2 Symmetric Multiprocessing"



The screenshot shows the IBM Navigator for i interface. The 'Software' section is active, displaying a table of installed software. The 'Description' column is filtered to show 'Db2'. The table contains one row for 'DB2 Symmetric Multiprocessing'.

Product ID	Option	Feature	Installed	Release Level	Description	Type	Compatible	State
5770SS1	26	5107	YES	V7R4M0	DB2 Symmetric Multiprocessing	CODE	YES	INSTALLED

Filtered Rows: 1 | Total Rows: 223

## [IBM i Portfolio Simplification](#)

# Db2 Symmetric Multiprocessing

## IBM i Simplification, *Stage 2: October 10, 2023 Announcement* Let's Create

- Stage 1 - [June 1, 2022 announcement](#): Selected LPPs and IBM i features priced at \$0
  - Until Stage 2, LPPs and features still needed to be ordered, keys applied (if applicable), etc.
- Stage 2 - [October 10, 2023 announcement](#): IBM i License Manager changes
  - Install PTF: no ordering, no license checking, software keys not required

### Products/Options \$0: keys required prior to 10/24/2023

- InfoPrint Server (5722-IP1)
- Advanced DBCS Printer Support (5761-AP1)
- Communications Utilities (5761-CM1)
- Advanced Job Scheduler (5770-JS1)
- Performance Tools (5770-PT1)
- Query for i (5770-QU1)
- Db2 Query Mgr & SQL Dev Kit (5770-ST1)
- IBM i Access Family (5770-XW1)

- IBM i Optional Features: (5770-SS1)
- Media & Storage Extension – Opt 18
  - PSF 1-55 IPM Support – Opt 36
  - PSF 1-100 IPM Support – Opt 37
  - PSF AnySpeed – Opt 38
  - HA Switchable Resource Opt 41
  - HA Journal Performance – Opt 42

### Products/Options \$0: not keyed

- Rational Application Management Toolset (5770-AMT)
- AFP Font Collection (5733-B45)
- AFP DBCS Fonts (5769-FN1)
- AFP Fonts (5769-FNT)
- XML Toolkit (5733-XT2)

### IBM i Optional Features:

- **Db2 Symmetric Multiprocessing – Opt 26**
- Db2 Multisystem – Opt 27

# Db2 Symmetric Multiprocessing

“I have only 1 IBM i core license so SMP will not help”

The image shows two overlapping windows from a Db2 monitoring interface. The left window, titled 'Processors', displays usage metrics and processor information. The right window, titled 'Environment Information for SQL Statement', shows various system parameters.

**Processors**

Total usage (elapsed)	5.2%
Interactive performance	100%
Shared processor pool usage (elapsed)	12.2%
Uncapped CPU capacity pool usage(elapsed)	2.6%

**Processor Information**

Type of processors	UNCAPPED
Processing power	0.5
Virtual processors	2

**Environment Information for SQL Statement**

Memory Pool Size	3.995E10
Memory Pool ID	*BASE
Share of Memory Available(bytes)	9.081E8
Average Active Used	44
Maximum Active Threads Allowed in Pool	2,000
Average Active In The Pool	44
Number of Processors	2 (16)
Workload Group Specified	No
Processor Units	.5
Date format	ISO
Date separator	-
Time format	ISO

# Db2 Symmetric Multiprocessing

## [Introduction to DB2 Symmetric Multiprocessing for IBM i](#)

## [SQE Symmetric Multiprocessing Changes](#)

On IBM® i 7.5, the behavior of the **PARALLEL\_DEGREE \*OPTIMIZE** setting is changed for database queries.

When Db2 Symmetric Multiprocessing (SMP) is enabled, the **PARALLEL\_DEGREE \*OPTIMIZE** enables the query optimizer to determine the optimal number of parallel tasks with which to run a query.

In most environments, SMP provides the greatest benefit to longer-running queries. With this update, Db2 for i now offers a feature which enables the optimizer to only consider SMP for queries that run for longer than a user-configurable time threshold. This allows the optimizer to apply SMP specifically to those queries that will benefit the most. When running with **PARALLEL\_DEGREE \*OPTIMIZE** or **\*OPTIMIZE%**, all queries will initially run without the use of SMP. Only when queries have exceeded the time threshold will they be eligible to use SMP. This feature is delivered with the Db2 for i 7.5 PTF group SF99950 level 3.

### **Minimum Runtime for Parallel Queries**

A new QAQQINI option, **PARALLEL\_MIN\_TIME**, describes the minimum time in seconds that a query must run before the optimizer may consider using SMP. When **PARALLEL\_DEGREE \*OPTIMIZE** is used, queries will initially execute with a non-parallel implementation. If a query completes within the configured **PARALLEL\_MIN\_TIME** number of seconds, the optimizer will not run the query with SMP. If the query runs longer than the configured amount of time, the optimizer may choose to replace the plan with a parallel implementation. If a parallel implementation is chosen, all subsequent executions of the query will also use SMP. The optimizer retains information about the choice of SMP for each query (identified by its QRO hash) until the plan cache is cleared or until the next IPL.

By default, **PARALLEL\_MIN\_TIME** is set to 60 seconds, and may be set to any integer value between 2 and 100,000, inclusive. In order for **PARALLEL\_MIN\_TIME** to take effect, **PARALLEL\_DEGREE** must be set to **\*OPTIMIZE** or **\*OPTIMIZE %**.

The new behavior does not apply to native queries, when Adaptive Query Processing is disabled, or when **PARALLEL\_MIN\_TIME** is set to **\*NONE**. Under these conditions, all queries using **PARALLEL\_DEGREE \*OPTIMIZE** will consider parallel execution during their initial and all subsequent optimizations, which is the same behavior as prior releases.

# Db2 Symmetric Multiprocessing

## [Creating the QAQQINI query options file](#)

### Creating the QAQQINI query options file

Last Updated: 2025-10-07

Each system is shipped with a QAQQINI template file in schema QSYS. The QAQQINI file in QSYS is to be used as a template when creating all user specified QAQQINI files.

To create your own QAQQINI file, use the **Create Duplicate Object (CRTDUPOBJ)** command. Create a copy of the QAQQINI file in the schema specified on the **Change Query Attributes (CHGQRYA)** QRYOPTLIB parameter. The file name must remain QAQQINI. For example:

```
CRTDUPOBJ OBJ(QAQQINI)
          FROMLIB(QSYS)
          OBJTYPE(*FILE)
          TOLIB(MYLIB)
          DATA(*YES)
          TRG(*YES)
```

System-supplied triggers are attached to the QAQQINI file in QSYS therefore it is imperative that the only means of copying the QAQQINI file is through the CRTDUPOBJ CL command. If another means is used, such as **CPYF**, then the triggers could be corrupted. An error is signaled that the options file cannot be retrieved or that the options file cannot be updated.

Because of the trigger programs attached to the QAQQINI file, the following CPI321A informational message is displayed six times in the job log when the **CRTDUPOBJ** CL is used to create the file. These messages are not an error; they are only informational messages.

CPI321A Information Message: Trigger QSYS\_TRIG\_&1\_\_QAQQINI\_\_00000&N in library &1 was added to file QAQQINI in library &1. The ampersand variables (&1, &N) are replacement variables that contain either the library name or a numeric value.

**Note:** It is highly recommended that the file QAQQINI, in QSYS, not be modified. This file is the original template that is duplicated into QUSRSYS or a user specified library for use.

# Db2 Symmetric Multiprocessing

## [OVERRIDE\\_QAQQINI procedure](#)

### OVERRIDE\_QAQQINI procedure

Last Updated: 2025-10-07

The OVERRIDE\_QAQQINI procedure creates and modifies a temporary version of the QAQQINI file.

The temporary QAQQINI file will be created in QTEMP. It inherits all query options that are already in place for the job. The OVERRIDE\_QAQQINI procedure can be called multiple times to establish job specific QAQQINI settings.

The procedure can also be called to discard the temporary customization settings.

**Authorization:** For most QAQQINI options, none is required. For the following options, the caller must have \*JOBCTL special authority or be authorized to the QIBM\_DB\_SQLADM function usage ID. These options are more restrictive because they can affect the performance of other jobs.

- QUERY\_TIME\_LIMIT when the *option-value* is not 0.
- STORAGE\_LIMIT
- PARALLEL\_DEGREE when the *option-value* is not \*NONE.
- PARALLEL\_MAX\_SYSTEM\_CPU

# Db2 Symmetric Multiprocessing

## *override-option*

An integer value that indicates the function to perform.

**1**

Create the QAQQINI override file. A procedure call with this *override-option* value must be run before option 2 can be used to change QAQQINI options.

**2**

Set a QAQQINI option to the specified value. See [QAQQINI query options](#) for the list of options and values.

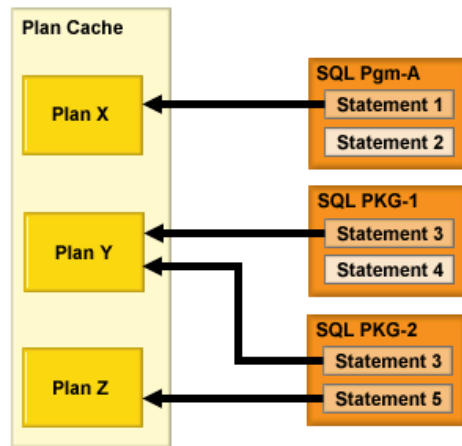
**3**

Discard the temporary QAQQINI file.

```
1 -- Create the QAQQINI override file
2 call qsys2.override_qaqqini(1);
3 -- Set a QAQQINI option to the specified value.
4 call qsys2.override_qaqqini(2, 'PARALLEL_MIN_TIME', '2');
5 -- Discard the temporary QAQQINI file.
6 call qsys2.override_qaqqini(3);
```

# Plan Cache Documentation

The following graphic shows the concept of re-usability of the query access plans stored in the plan cache:



As shown in the previous graphic, statements from packages and programs are stored in unique plans in the plan cache. If Statement 3 exists in both SQL package 1 and SQL package 2, the plan is stored once in the plan cache. The plan cache is interrogated each time a query is executed. If an access plan exists that satisfies the requirements of the query, it is used to implement the query. Otherwise a new access plan is created and stored in the plan cache for future use.

The plan cache is automatically updated with new query access plans as they are created. When new statistics or indexes become available, an existing plan is updated the next time the query is run. The plan cache is also automatically updated by the database with runtime information as the queries are run.

⌞ Each plan cache entry contains the original query, the optimized query access plan, and cumulative runtime information gathered during the runs of the query. The size of these objects depends largely on the complexity of the SQL statements that are being executed. It is the size of these objects that is counted when calculating the plan cache size. ⌋

# Plan Cache Documentation

↳ The size of the plan cache may be controlled by either a system-managed auto-sizing algorithm or by an explicitly set value. By default, the system will use automatic sizing, but this can be disabled by setting the SQL Plan Cache Threshold Size. See the SQL plan cache properties topic for more information: [SQL plan cache properties](#) >|

↳ When auto-sizing is enabled, the system will begin after each IPL with a default plan cache size of 512 MB. As queries run, the associated plans are placed into the plan cache. Once the plan cache exceeds the current limit set by auto-sizing, the system will evaluate the current hit ratio for the cache. This is the ratio of the number of times a query could re-use a plan from the cache compared to the number of times a query was run. For example, if ten queries were run and nine of the queries used plans from the plan cache while the other one required a full optimization, the hit ratio is 90%. If the current hit ratio equals or exceeds the SQL Plan Cache Target Ratio ( see the SQL plan cache properties topic for more information: [SQL plan cache properties](#)), the system considers the plan cache to be operating at an efficient size and will not adjust the size of the cache. >|

↳ If the current hit ratio is below the target for the plan cache and no other storage constraints exist, the system will automatically increase (up to the Maximum Plan Cache Size for Auto-sizing) the size of the plan cache. The plan cache will grow incrementally, allowing more plans to be stored, until the current hit ratio meets the target ratio. >|

↳ Once the target hit ratio has been achieved, the determined size of the cache is maintained by an automatically scheduled background task. The purpose of this task is to go through the plan cache and to remove plans when the cache grows too large. This task will remove access plans based upon age, how frequently it is used, and how much cumulative resources (CPU/IO) were consumed. >|

↳ Each plan cache entry may also have query runtime objects associated with it. These runtime objects are the real executable objects and temporary storage containers (hash tables, sorts, temporary indexes, and so on) used to run the query. Although these objects are not included in the plan cache size calculation, they may indirectly affect and be affected by the plan cache size. This is because, in addition to honoring the determined plan cache size, the system also seeks to keep the total temporary storage usage for inactive, cached plans within an internally determined threshold. Unlike the plan cache size calculation, this temporary storage calculation considers both the cached plans and the associated runtime objects. If this temporary storage calculation exceeds a system determined percentage of the system auxiliary storage pool (ASP) or if the system storage lower limit (defined by QSTGLOWLMT) is surpassed, the system considers the plan cache to be using excessive temporary storage. >|

↳ To reduce the excessive temporary storage, the automatically scheduled background task will begin by removing runtime objects from cached plans. This will continue until the plan cache's temporary storage usage is within the system's constraints, and this will happen even if all the cached plan entries themselves fit within the determined plan cache size. If temporary storage usage remains excessive even after the runtime objects are removed, the system will begin a process of incrementally reducing the plan cache size and will continue this reduction until it reaches the minimum value of 512 MB or until the plan cache temporary storage usage is no longer excessive. >|

# Plan Cache Documentation

If the current hit ratio is below the target for the plan cache and no other storage constraints exist, the system will automatically increase (up to the Maximum Plan Cache Size for Auto-sizing) the size of the plan cache. The plan cache will grow incrementally, allowing more plans to be stored, until the current hit ratio meets the target ratio.

Once the target hit ratio has been achieved, the determined size of the cache is maintained by an automatically scheduled background task. The purpose of this task is to go through the plan cache and to remove plans when the cache grows too large. This task will remove access plans based upon age, how frequently it is used, and how much cumulative resources (CPU/IO) were consumed.

Each plan cache entry may also have query runtime objects associated with it. These runtime objects are the real executable objects and temporary storage containers (hash tables, sorts, temporary indexes, and so on) used to run the query. Although these objects are not included in the plan cache size calculation, they may indirectly affect and be affected by the plan cache size. This is because, in addition to honoring the determined plan cache size, the system also seeks to keep the total temporary storage usage for inactive, cached plans within an internally determined threshold. Unlike the plan cache size calculation, this temporary storage calculation considers both the cached plans and the associated runtime objects. If this temporary storage calculation exceeds a system determined percentage of the system auxiliary storage pool (ASP) or if the system storage lower limit (defined by QSTGLOWLMT) is surpassed, the system considers the plan cache to be using excessive temporary storage.

To reduce the excessive temporary storage, the automatically scheduled background task will begin by removing runtime objects from cached plans. This will continue until the plan cache's temporary storage usage is within the system's constraints, and this will happen even if all the cached plan entries themselves fit within the determined plan cache size. If temporary storage usage remains excessive even after the runtime objects are removed, the system will begin a process of incrementally reducing the plan cache size and will continue this reduction until it

# Plan Cache Documentation

## IBM i temporary storage percentage

### Question

What are some options to determine what my temporary storage percentage on the IBM i?

### Answer

Here are four options to check your temporary storage percentage. There may be others you like better as well.

1) Using SQL (simple calculation, modify / add precision as you'd like)

Use [Run SQL Scripts](#) (part of [Access Client Solutions](#))

```
SELECT CURRENT_TEMPORARY_STORAGE,
       SYSTEM_ASP_STORAGE,
       (CURRENT_TEMPORARY_STORAGE * 100)
       / SYSTEM_ASP_STORAGE as Temp_Percent
FROM QSYS2.SYSTEM_STATUS_INFO
;
```

CURRENT_TEMPORARY_STORAGE	SYSTEM_ASP_STORAGE	TEMP_PERCENT
6790	286331	2

2) Database specific

Use [IBM i command line](#)

- CALL PGM(QQ0000CACH) PARM('F:QTEMP/UFILE:U')
- runqzyc () qtemp/ufile
- Temp % is on the first screen

```
Temp bytes - 6763470848 (6 GB)
Temp % - 2.36
```

d. **bonus:** page down to SQE ACTIVE CURSOR SUMMARY to see Job with largest SQE temp usage

```
SQE ACTIVE CURSOR SUMMARY
Total SQE cursors on the system - 23
Total SQE ROQ sizes on the system - 1832120 (1 MB)
Job with largest SQE temp usage - QDBSRV04 QSYS 132869 used 838344 bytes.
```

# Plan Cache Documentation

## IBM i temporary storage percentage

3) An IBM i API - (message can be confusing however)

Use [IBM i command line](#)

a. CALL [QWCCTLTS](#) PARM(\*DSPLMT \*ALL)

b. F1 on the Temporary storage threshold reached message.

Cause . . . . . : The amount of storage used for temporary objects in the system ASP is **2.3621** percent.

4) An IBM i API - Add functionality to WRKSYSSTS

Use [IBM i command line](#)

a. CALL [QWCCTLTS](#) PARM(\*SETLMPCT \*ALL 80)

b. [WRKSYSSTS](#)

c. F19=Extended system status

```
% temporary storage used . . . . . :      2.3
```

```
% temporary storage limit . . . . . :      80
```

d. note that the 80 is from step 4.a.

e. To remove a warning limit for all temporary storage on the system (and set it back to the default of no warning limit)

```
CALL QWCCTLTS PARM(*SETLMPCT *ALL 0)
```

# Plan Cache

## Using Run SQL Scripts

```
1 CL:CALL PGM(QQQOOOCACH) PARM('F:QTEMP/UFILE:U');  
2  
3 Select * from qtemp.ufile;
```

SQE Temp Storage Threshold %	- 7.00
Using new temp threshold calculation (SQE Plancache Temp Usage)	
New Temp Threshold Calculation %	- 2.18
Old Temp Threshold Calculation %	- 6.69
Using new autosize storage calculation	
Original Autosize Calculation %	- 18.64
New Autosize Calculation %	- 2.18

# Plan Cache

Remember this section for link before?

Use [IBM i command line](#)

- a. `CALL PGM(QQQ000CACH) PARM('F:QTEMP/UFILE:U')`
- b. `runqry () qtemp/ufile`
- c. Temp % is on the first screen

```
Temp bytes      - 6763470848  (6 GB)
Temp %          - 2.36
```

# Plan Cache

If Autosizing is limited due to Temporage Storage by the 7 % rule are:

Change the Plan Cache Configuration by switching off Auto Sizing

The screenshot shows the 'Plan Cache Properties' dialog box with the 'Change Plan Cache Configuration' sub-dialog open. The sub-dialog has the following options:

- \*SAME (\*AUTO)
- \*DEFAULT (\*AUTO)
- Minimum value (50 MB)
- Maximum value (51200 MB)
- Specify a value (MB): 25,625

The main dialog shows various statistics for the Plan Cache, including:

- Current Number of Plans in Cache: 1854
- Total Number of Plans Built Since Start: 28644
- Current Plan Cache Size: 2923 MB
- Current Plan Cache Size Threshold: \*AUTO
- Maximum Plan Cache Size For Autosizing: \*DEFAULT (3072) MB
- Target Plan Cache AutoSize Ht Ratio: 82 %
- Target Plan Cache AutoSize Ht Ratio: \*DEFAULT (90) %

# Plan Cache

Switching off Auto Sizing means monitoring - What is the alternative?

Ask IBM to make the 7 % rule flexibel?

Well that is what I did by entering an IBM idea

[Make it possible to change the SQE Temp Storage 7% Threshold](#)

If you think this is a good idea you have the option to vote for it 🧐

## Make it possible to change the SQE Temp Storage 7% Threshold

👍 3 · 💬 1 · ⌚ Created on 12 Nov 2025 · Future consideration · +

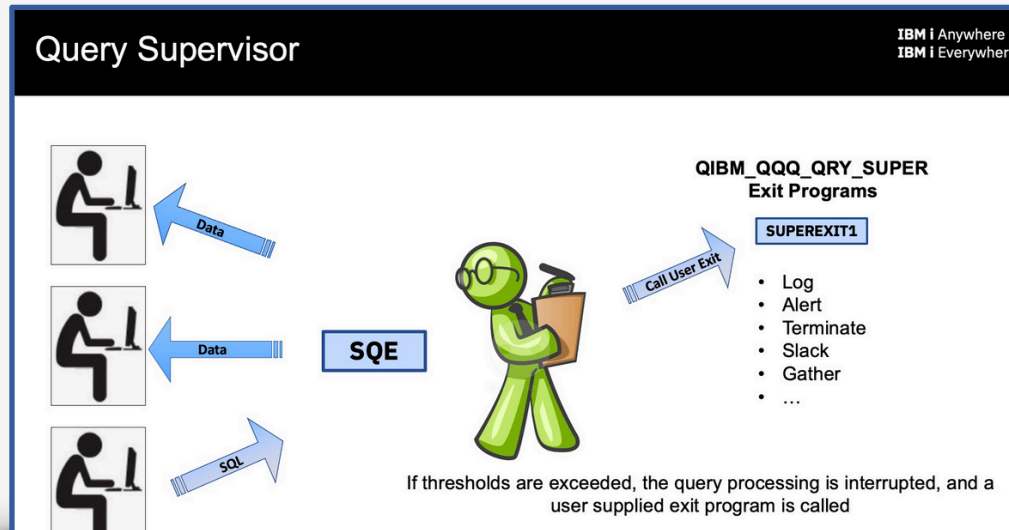
The current fixed SQE Temp Storage Threshold is set to 7%. I have no idea how long this value is in place, but my guess is that values was set to this values decades ago. Over the year the usage of temporary storage has grown on nearly every IBM i LPAR I have seen. Also the use of the IBM i Database has grown.

With that in mind when looking at the plan cache details using the ACS SQL Performance Center I regularly see a date behind the the "Last Autosizing Limited Due to Temporary Storage" cased by the 7% rule. In some case the LPAR has a disk usage 80 to 90%, but in many cases this means that there is still an abundance of free disk space left.

I know that the Plan Cache allows you to change the Plan Cache Configuration and change the Current Plan Cache Size Threshold form the default being \*AUTO to a value in MB. The downside of this that after you made the change you need to monitor the size of the Plan Cache from the moment you so forever. With that in mind the value \*AUTO is the better option I think. If only IBM would make it possible to change that 7 % threshold ;-)

This idea is all about adding flexibility to the 7 % threshold in allowing us to change it.

[IBM i 7.3 TR10](#) and [7.4 TR4](#) on April 13 2021



### Db2 for i - Services (new)

- [QSYS2.ADD\\_QUERY\\_THRESHOLD](#)
- [QSYS2.END\\_IDLE\\_SQE\\_THREADS](#)
- [QSYS2.QUERY\\_SUPERVISOR](#)
- [QSYS2.REMOVE\\_QUERY\\_THRESHOLD](#)

## Where the journey started

### ✓ Abstract

The Db2 for i SQL Query Engine (SQE) provides a Query Supervisor which enables real-time monitoring of resource consumption by SQL and native queries.

### ✓ Documentation

[Query Supervisor](#)

### ✓ For complete details

[Query Supervisor](#)

[ADD\\_QUERY\\_THRESHOLD procedure](#)

[REMOVE\\_QUERY\\_THRESHOLD procedure](#)

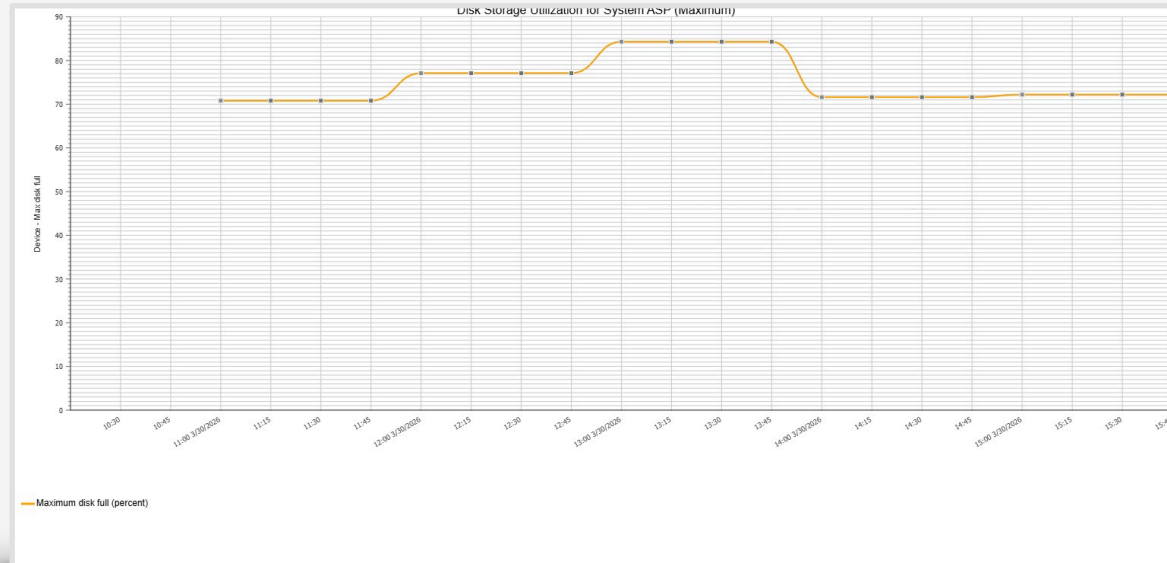
[QUERY\\_SUPERVISOR view](#)

[Query Supervisor Exit Program](#)

(updated to include CL, to join RPG and C, versions of sample exit programs)

## Potential Situation

## This could happen to you or not?

✓ **Disk space usage 90 %**

After investigation afterwards!!!

- An ad hoc SQL query was initiated (in this case from an Excel sheet)
- The user did “cancel” the query
- In the background the query continued
- An automated trigger process “killed” all ODBC tasks QZDASOINIT

✓ **A rough remedy**

Nothing in place to detect the real cause.

✓ **Tailor-made solution**

The Query Supervisor is part of this solution.

# Three Parts

## The Theory

01

What is the Query Supervisor

---

- = The thresholds
- = The triggers

## The Explanation

02

How does the Query Supervisor work

---

- = Components
- = How to activate?
- = How to review?
- = How to check?

## The Building

03

How to build and implement the Query Supervisor

---

- = Schemas
- = Tables
- = Procedures
- = Programs
- = Bring it all together
- = Testing

# The Theory

## The Threshold

When running SQL statements they:

---

- Have an I/O count
- Use CPU time.
- Run for an amount of time
- Use temporary storage

How to determine the thresholds?

What is “too much”?

The answer as always is “it depends”.

What can we do to get insight?

Daily Plan cache Management.

# Daily Plan Cache Management

The screenshot shows an SQL IDE window titled 'Untitled 1'. The main editor area displays the following SQL code:

```
-- category: Db2 for i Services
-- description: daily SQL Plan Cache management

CL: CRTLIB SNAPSHOTS;
CL: CRTLIB EVENTMONS;
-- Purpose: This procedure captures detail on SQL queries.
-- 1) The 100 most expensive SQL queries are captured into a SQL Plan Cache Snapshot named SNAP
-- 2) An SQL Plan Cache Event Monitor is started using a name SNAPSHOTS/EVT<julian-date>. The p
-- 3) For both 1 & 2, only the 14 most recent days are kept online.
-- 4) For both 1 & 2, the new monitor and snap shot are imported into the IBM i Access Client S
CREATE OR REPLACE PROCEDURE SNAPSHOTS.DAILY_PC_MANAGEMENT ()
LANGUAGE SQL
BEGIN
DECLARE not_found CONDITION FOR '02000';
DECLARE SNAP_NAME CHAR(10);
DECLARE OLDEST_SNAP_NAME CHAR(10);
DECLARE SNAP_COMMENT VARCHAR(100);
DECLARE EVENT_MONITOR_NAME CHAR(10);
DECLARE YESTERDAY_EVENT_MONITOR_NAME CHAR(10);
DECLARE OLDEST_EVENT_MONITOR_NAME CHAR(10);
DECLARE OLD_EVENT_MONITOR_ID CHAR(10);
DECLARE v_not_found BIGINT DEFAULT 0;

-- A Julian date is the integer value representing a number of days
-- from January 1, 4713 B.C. (the start of the Julian calendar) to
-- the date specified in the argument.
SET SNAP_NAME = 'SNP' CONCAT JULIAN_DAY(current date);
SET OLDEST_SNAP_NAME = 'SNP' CONCAT JULIAN_DAY(current date - 14 days);
SET EVENT_MONITOR_NAME = 'EVT' CONCAT JULIAN_DAY(current date);
SET OLDEST_EVENT_MONITOR_NAME = 'EVT' CONCAT JULIAN_DAY(current date - 14 days);
SET YESTERDAY_EVENT_MONITOR_NAME = 'EVT' CONCAT JULIAN_DAY(current date - 1 day);
-----
-- Process the Top 100 most expensive queries
-----
-- Capture the topN queries and import the snapshot
CALL QSYS2.DUMP_PLAN_CACHE_topN('SNAPSHOTS', SNAP_NAME, 100);

-- Remove the oldest TOPN snapshot
BEGIN
DECLARE CONTINUE HANDLER FOR not_found
SET v_not_found = 1;
CALL QSYS2.REMOVE_PC_SNAPSHOT('SNAPSHOTS', OLDEST_SNAP_NAME);
```

The IDE interface includes a toolbar at the top, a left sidebar with a file explorer showing 'Untitled 1', and a bottom status bar indicating 'Connected to relational database Testcbuf'. The 'SQL Examples' panel on the right shows a list of examples, with 'daily' selected. A red arrow points from the 'SQL' icon in the toolbar to the 'SQL Examples' panel.

# Daily Plan Cache Management

SQL Performance Center - TESTCBU

File Edit View Actions Tools Help

Testcbu Plan Cache Statements Index Advisor Maintained Temporary Indexes Index Evaluator Active Query Info SQL Details for Jobs

Plan Cache Performance Monitors Plan Cache Snapshots Plan Cache Event Monitors

Testcbu ▶ Plan Cache Snapshots

Name	Schema	Table	Created By	Date Created
SNAPSHOTS ASP_2605100510235502	SNAPSHOTS	ASP_260510	HA	05/10/2026 11:55:02 PM
SNAPSHOTS ASP_2605110511235501	SNAPSHOTS	ASP_260511	HA	05/11/2026 11:55:01 PM
SNAPSHOTS ASP_2605120512235502	SNAPSHOTS	ASP_260512	HA	05/12/2026 11:55:02 PM
SNAPSHOTS ASP_2605130513235502	SNAPSHOTS	ASP_260513	HA	05/13/2026 11:55:02 PM
SNAPSHOTS ASP_2605140514235502	SNAPSHOTS	ASP_260514	HA	05/14/2026 11:55:02 PM
SNAPSHOTS ASP_2605150515235501	SNAPSHOTS	ASP_260515	HA	05/15/2026 11:55:01 PM
SNAPSHOTS ASP_2605160516235502	SNAPSHOTS	ASP_260516	HA	05/16/2026 11:55:02 PM
SNAPSHOTS ASP_2605170517235502	SNAPSHOTS	ASP_260517	HA	05/17/2026 11:55:02 PM
SNAPSHOTS ASP_2605180518235502	SNAPSHOTS	ASP_260518	HA	05/18/2026 11:55:02 PM
SNAPSHOTS ASP_2605190519235503	SNAPSHOTS	ASP_260519	HA	05/19/2026 11:55:03 PM
SNAPSHOTS ASP_2605200520235504	SNAPSHOTS	ASP_260520	HA	05/20/2026 11:55:04 PM
SNAPSHOTS ASP_2605210521235505	SNAPSHOTS	ASP_260521	HA	05/21/2026 11:55:05 PM
SNAPSHOTS ASP_2605220522235505	SNAPSHOTS	ASP_260522	HA	05/22/2026 11:55:05 PM
SNAPSHOTS ASP_2605230523235506	SNAPSHOTS	ASP_260523	HA	05/23/2026 11:55:06 PM
SNAPSHOTS SYS_2605100510235502	SNAPSHOTS	SYS_260510	RUDI	05/10/2026 11:55:02 PM
SNAPSHOTS SYS_2605110511235503	SNAPSHOTS	SYS_260511	RUDI	05/11/2026 11:55:03 PM
SNAPSHOTS SYS_2605120512235502	SNAPSHOTS	SYS_260512	RUDI	05/12/2026 11:55:02 PM
SNAPSHOTS SYS_2605130513235502	SNAPSHOTS	SYS_260513	RUDI	05/13/2026 11:55:02 PM
SNAPSHOTS SYS_2605140514235502	SNAPSHOTS	SYS_260514	RUDI	05/14/2026 11:55:02 PM
SNAPSHOTS SYS_2605150515235501	SNAPSHOTS	SYS_260515	RUDI	05/15/2026 11:55:01 PM
SNAPSHOTS SYS_2605160516235501	SNAPSHOTS	SYS_260516	RUDI	05/16/2026 11:55:01 PM
SNAPSHOTS SYS_2605170517235500	SNAPSHOTS	SYS_260517	RUDI	05/17/2026 11:55:00 PM
SNAPSHOTS SYS_2605180518235501	SNAPSHOTS	SYS_260518	RUDI	05/18/2026 11:55:01 PM
SNAPSHOTS SYS_2605190519235501	SNAPSHOTS	SYS_260519	RUDI	05/19/2026 11:55:01 PM
SNAPSHOTS SYS_2605200520235501	SNAPSHOTS	SYS_260520	RUDI	05/20/2026 11:55:01 PM
SNAPSHOTS SYS_2605210521235500	SNAPSHOTS	SYS_260521	RUDI	05/21/2026 11:55:00 PM
SNAPSHOTS SYS_2605220522235500	SNAPSHOTS	SYS_260522	RUDI	05/22/2026 11:55:00 PM
SNAPSHOTS SYS_2605230523235501	SNAPSHOTS	SYS_260523	RUDI	05/23/2026 11:55:01 PM

# Daily Plan Cache Management

Testcbu ► Plan Cache Snapshots

Name	Schema	Table	Created By	Date Created
SNAPSHOTS ASP_2605100510235502	SNAPSHOTS	ASP_260510	HA	05/10/2026 11:55:02 PM
SNAPSHOTS ASP_2605110511235501	SNAPSHOTS	ASP_260511	HA	05/11/2026 11:55:01 PM
SNAPSHOTS ASP_2605120512235502	SNAPSHOTS	ASP_260512	HA	05/12/2026 11:55:02 PM
SNAPSHOTS ASP_2605130513235502	SNAPSHOTS	ASP_260513	HA	05/13/2026 11:55:02 PM
SNAPSHOTS ASP_2605140514235502	SNAPSHOTS	ASP_260514	HA	05/14/2026 11:55:02 PM
SNAPSHOTS ASP_2605150515235501	SNAPSHOTS	ASP_260515	HA	05/15/2026 11:55:01 PM
SNAPSHOTS ASP_2605160516235502	SNAPSHOTS	ASP_260516	HA	05/16/2026 11:55:02 PM
SNAPSHOTS ASP_2605170517235502	SNAPSHOTS	ASP_260517	HA	05/17/2026 11:55:02 PM
SNAPSHOTS ASP_2605180518235502	SNAPSHOTS	ASP_260518	HA	05/18/2026 11:55:02 PM
SNAPSHOTS ASP_2605190519235503	SNAPSHOTS	ASP_260519	HA	05/19/2026 11:55:03 PM
SNAPSHOTS ASP_2605200520235504	SNAPSHOTS	ASP_260520	HA	05/20/2026 11:55:04 PM
SNAPSHOTS ASP_2605210521235505	SNAPSHOTS	ASP_260521	HA	05/21/2026 11:55:05 PM
SNAPSHOTS ASP_2605220522235505	SNAPSHOTS	ASP_260522	HA	05/22/2026 11:55:05 PM
SNAPSHOTS ASP_2605230523235506	SNAPSHOTS	ASP_260523	HA	05/23/2026 11:55:06 PM
SNAPSHOTS SYS_2605100510235502	SNAPSHOTS	SYS_260510	RUDI	05/10/2026 11:55:02 PM
SNAPSHOTS SYS_2605110511235503	SNAPSHOTS	SYS_260511	RUDI	05/11/2026 11:55:03 PM
SNAPSHOTS SYS_2605120512235502	SNAPSHOTS	SYS_260512	RUDI	05/12/2026 11:55:02 PM
SNAPSHOTS SYS_2605130513235502	SNAPSHOTS	SYS_260513	RUDI	05/13/2026 11:55:02 PM
SNAPSHOTS SYS_2605140514235502	SNAPSHOTS	SYS_260514	RUDI	05/14/2026 11:55:02 PM
SNAPSHOTS SYS_2605150515235501	SNAPSHOTS	SYS_260515	RUDI	05/15/2026 11:55:01 PM
SNAPSHOTS SYS_2605160516235501	SNAPSHOTS	SYS_260516	RUDI	05/16/2026 11:55:01 PM
SNAPSHOTS SYS_2605170517235500	SNAPSHOTS	SYS_260517	RUDI	05/17/2026 11:55:00 PM
SNAPSHOTS SYS_2605180518235501	SNAPSHOTS	SYS_260518	RUDI	05/18/2026 11:55:01 PM
SNAPSHOTS SYS_2605190519235501	SNAPSHOTS	SYS_260519	RUDI	05/19/2026 11:55:01 PM
SNAPSHOTS SYS_2605200520235501	SNAPSHOTS	SYS_260520	RUDI	05/20/2026 11:55:01 PM
SNAPSHOTS SYS_2605210521235500	SNAPSHOTS	SYS_260521	RUDI	05/21/2026 11:55:00 PM
SNAPSHOTS SYS_2605220522235500	SNAPSHOTS	SYS_260522	RUDI	05/22/2026 11:55:00 PM
SNAPSHOTS SYS_2605230523235501	SNAPSHOTS	SYS_260523	RUDI	05/23/2026 11:55:01 PM

# Query Supervisor limitations & scope

The Query Supervisor enables monitoring and management of queries that reach pre-determined thresholds of resource consumption. Unlike the Predictive Query Governor, which intervenes on the basis of *estimated* resource usage *before* a query runs, the Query Supervisor reacts to *actual* resource usage *while* the query runs.

The Query Supervisor monitors any SQL queries that are run to implement user SQL statements, including native database queries that are processed by SQE. SQL statements that do not require query processing such as a simple INSERT (for example, INSERT INTO <table> VALUES(123) ) or COMMIT are not monitored by the Query Supervisor. Query supervision does not occur for queries initiated by the IBM® i operating system or for SQL that has been classified as specific to operating system processing.

The Query Supervisor allows a Database Engineer (DBE) to:

- Deploy real-time notification solutions when a query exceeds a specific resource threshold
- Take actions to terminate queries based on real-time query consumption of system resources
- Capture and log performance details for long-running queries

By configuring the Query Supervisor, a DBE can proactively manage excessive resource usage, monitor for unexpected workload variation, and automatically end run-away queries. The Query Supervisor provides the infrastructure for establishing and detecting thresholds. The specific action(s) taken when a threshold is reached is determined by one or more exit programs that the DBE provides.

# The Theory

## The Triggers

Action to be taken when a threshold is exceeded?

---

- Ability to investigate the SQL statement
- End or Hold the SQL Statement
- Get notified
- Automate the action

Maximum of flexibility:





- Using custom build software
- You determine, but IBM offers example software

# The Explanation

## Components

Thresholds

---

-  Define Threshold
-  Maintain the Thresholds
-  Active the Thresholds
-  Remove the Threshold

## [Query Supervisor configuration and operation](#)

Query Supervisor thresholds are configured using the add and remove procedures:

[ADD\\_QUERY\\_THRESHOLD procedure](#)

[REMOVE\\_QUERY\\_THRESHOLD procedure](#).

The [QUERY\\_SUPERVISOR view](#) shows the defined thresholds.

The four threshold types which can be monitored by the Query Supervisor are:





1. **CPU Time** – The total processing unit time used by the query, in seconds
2. **Elapsed Time** – The total clock time, in seconds
3. **Temporary storage** – The amount of storage, in megabytes (MB), that the query allocates
4. **Total I/O count** – The total number of I/O operations

# The Explanation

## Components

Exit Point Programs

---

-  Query Supervisor Exit Program
-  Add Query Supervisor Exit Program
-  Remove Query Supervisor Exit Program
-  Review Query Supervisor Exit Program

## [Query Supervisor Exit Program](#)

The Query Supervisor exit program is called when a job is running a query and a resource threshold defined by the Query Supervisor is met or exceeded. This exit is called in the thread that is running the query.





The resource thresholds monitored by the Query Supervisor may be configured by using the [ADD\\_QUERY\\_THRESHOLD](#) procedure and the [REMOVE\\_QUERY\\_THRESHOLD](#) procedure.

```
SELECT *  
  FROM qsys2.EXIT_PROGRAM_INFO  
 WHERE exit_point_name LIKE 'QIBM_XXX_QRY_SUPER';
```

# The Building

## Schemas

Why a SQL Query Threshold table?

-  Flexibility
-  Transparency
-  Security
-  Traceability

SUPERVISOR.SQL\_QUERY\_THRESHOLD - TESTCBUF [redacted] (Testcbuf)

Table Columns Key Constraints Foreign Key Constraints Check Constraints Materialized Query Partitioning

Column Name	System Name	Data Type	Length	Nullable	Generated Value	Default Value	Hidden	Text	CCSID	Field
ACTIVE	ACTIVE	CHARACTER	4	No		'*NO'		Record Active Indicator	37	
THRESHOLD_NAME	TRSHLDNAME	CHARACTER	30	Yes		''		Threshold Name	37	
THRESHOLD_TYPE	TRSHLDTYPE	CHARACTER	17	Yes		Null		Threshold Type	37	
THRESHOLD_VALUE	TRSHLDVAL	INTEGER		Yes		0		Threshold Value		
JOB_NAMES	JOB_NAMS	VARCHAR	1,100	No		'*ALL'		Job Names	37	
INCLUDE_USERS	INCUSERS	CHARACTER	1,100	No		'*ALL'		Include Users	37	
EXCLUDE_USERS	EXCUSERS	CHARACTER	1,100	No		'*NONE'		Exclude Users	37	
SUBSYSTEMS	SBS_NAME	CHARACTER	1,100	No		'*ALL'		Subsystem Names	37	
DETECTION_FREQUENCY	FREQUENCE	INTEGER		No		600		Detection Frequency in seconds		
LONG_COMMENT	*COMMENT*	VARCHAR	2,000	No		''		Long Comment	37	

```

8 SELECT *
9 FROM supervisor.SQL_QUERY_THRESHOLD for update ;
10 stop;
    
```

ACTIVE	THRESHOLD_NAME	THRESHOLD_TYPE	THRESHOLD_VALUE	JOB_NAMES	INCLUDE_USERS	EXCLUDE_
*YES	The total cpu time, in sec.	CPU TIME	1,800	*ALL	*ALL	*NONE
*YES	The total clock time, in sec.	ELAPSED TIME	3,600	*ALL	*ALL	*NONE
*NO	The amount of storage, in MB.	TEMPORARY STORAGE	500	*ALL	*ALL	*NONE
*NO	The total number of I/O ops.	TOTAL IO COUNT	1,000,000,000	*ALL	*ALL	*NONE
*YES	The total clock time, rudi	ELAPSED TIME	60	QZDASSINIT	RUDI	*NONE

# SQL\_QUERY\_THRESHOLD Table

Field definition based on [ADD\\_QUERY\\_THRESHOLD procedure](#)

**threshold-name**

A character or graphic string that provides a name for the threshold. The name can be up to 30 characters long and can contain any characters including blanks. The name cannot be the same as an existing threshold name.

**threshold-type**

A character or graphic string that specifies the type of the threshold to be enforced.

**CPU TIME**

The total processing unit time used by the query, in seconds.

**ELAPSED TIME**

The total clock time, in seconds.

**TEMPORARY STORAGE**

The amount of storage, in megabytes (MB), that the query allocates.

**TOTAL IO COUNT**

The total number of I/O operations.

**threshold-value**

A big integer value that contains the threshold value, in units defined by the specified *threshold-type*. The value must be greater than 0.

**job-names**

A character or graphic string that specifies up to 100 job names separated by either a blank or a comma. Each job name can end with a wildcard character. For example, 'QPADEV\*' indicates that any job name starting with the characters 'QPADEV' is a match.

Only jobs running with a matching job name are eligible to be supervised. Can contain the following special value:





**\*ALL**

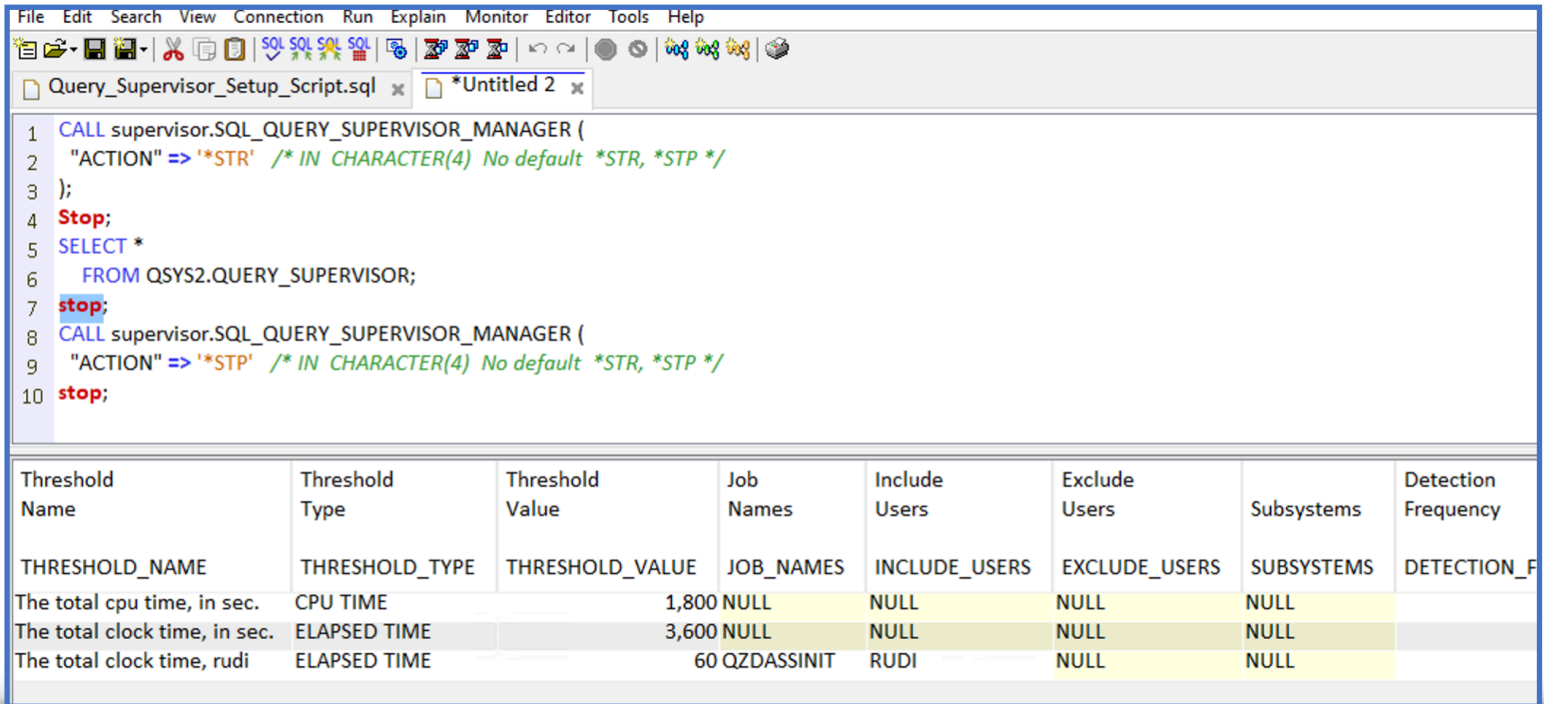
All jobs are supervised. This is the default.

# The Building

## Schemas

A SQL Procedure using the Query Threshold table

-  Flexibility
-  Transparency
-  Security
-  Traceability



```

1 CALL supervisor.SQL_QUERY_SUPERVISOR_MANAGER (
2  "ACTION" => '*STR' /* IN CHARACTER(4) No default *STR, *STP */
3 );
4 Stop;
5 SELECT *
6   FROM QSYS2.QUERY_SUPERVISOR;
7 stop;
8 CALL supervisor.SQL_QUERY_SUPERVISOR_MANAGER (
9  "ACTION" => '*STP' /* IN CHARACTER(4) No default *STR, *STP */
10 stop;
  
```

Threshold Name	Threshold Type	Threshold Value	Job Names	Include Users	Exclude Users	Subsystems	Detection Frequency
THRESHOLD_NAME	THRESHOLD_TYPE	THRESHOLD_VALUE	JOB_NAMES	INCLUDE_USERS	EXCLUDE_USERS	SUBSYSTEMS	DETECTION_F
The total cpu time, in sec.	CPU TIME	1,800	NULL	NULL	NULL	NULL	
The total clock time, in sec.	ELAPSED TIME	3,600	NULL	NULL	NULL	NULL	
The total clock time, rudi	ELAPSED TIME	60	QZDASSINIT	RUDI	NULL	NULL	

# SQL\_QUERY\_SUPERVISOR\_MANAGER Procedure

The screenshot displays the IBM Db2 SQL Editor interface for the procedure `SQL_QUERY_SUPERVISOR_MANAGER` in the `SUPERVISOR` schema. The main window shows the procedure details, and a smaller window shows the routine body code.

**Procedure Details:**

- Name: SQL\_QUERY\_SUPERVISOR\_MANAGER
- Schema: SUPERVISOR
- Specific name: SQS\_MGR
- Language: SQL
- Program or service program: SQS\_MGR
- Program type: Main (program)
- Definer: RUDI
- Program owner: RUDI
- Date created: 04/10/2026, 05:53:22 PM
- Last altered: Never
- SQL path at create time: "QSYS", "QSYS2", "SYSPROC", "SYSIBMADM"

**Parameters:**

Number	Mode	Name	Data Type	Length	CCSID	Locator	Default Value
1	IN	"ACTION"	CHARACTER	4			No default

**Routine Body:**

```

BEGIN
-----
DECLARE FULLCMD VARCHAR ( 6000 ) ;
DECLARE CONTINUE HANDLER FOR SQLEXCEPTION
BEGIN
-- CALL QSYS2.QCMDEXC(
-- 'SNDSMTPEMM RCP("rudi.vanhelvoirt@vanhelvoirt.nl") SUBJECT("Fout in procedure Start/Stop SQL Query Supervisor") NOTE("E.e.a. handmatig
controleren!")'
-- );
END ;
FOR READ AS SQL_QUERY_THRESHOLD_FILE_CURSOR CURSOR FOR
SELECT * FROM SUPERVISOR . SQL_QUERY_THRESHOLD WHERE ACTIVE = '*YES'
DO
IF ACTION = '*STR' THEN
SET FULLCMD = 'CALL QSYS2.ADD_QUERY_THRESHOLD(THRESHOLD_NAME => "" CONCAT TRIM ( THRESHOLD_NAME ) CONCAT "", THRESHOLD_TYPE
=> "" CONCAT TRIM ( THRESHOLD_TYPE ) CONCAT "",
THRESHOLD_VALUE => "" CONCAT TRIM ( THRESHOLD_VALUE ) CONCAT "", JOB_NAMES => "" CONCAT TRIM ( JOB_NAMES )
CONCAT "",
INCLUDE_USERS => "" CONCAT TRIM ( INCLUDE_USERS ) CONCAT "", EXCLUDE_USERS => "" CONCAT TRIM ( EXCLUDE_USERS )
CONCAT "",
SUBSYSTEMS => "" CONCAT TRIM ( SUBSYSTEMS ) CONCAT "", DETECTION_FREQUENCY => ' CONCAT CHAR (
DETECTION_FREQUENCY ) CONCAT ' ;

```

# The Building

## Exit Programs

Do not reinvent the wheel

---

- Get started
- Start learning
- Build confidence
- Use the [Query Supervisor example exit programs](#)

## Query Supervisor example exit programs

Last Updated: 2025-10-07

The exit programs in this section demonstrate some of the ways an exit program can be used when a Query Supervisor threshold is reached. They are provided to enable quick and easy adoption of Query Supervisor.

The details of the exit program interface can be found here: [Query Supervisor Exit Program](#)

- [Exit program to send message to QSYSOPR](#)  
This Query Supervisor exit program sends an informational message to the QSYSOPR message queue.
- [Exit program to end query](#)  
This Query Supervisor exit program shows how to request that the query that was running when the threshold was reached is to be stopped.
- [Exit program to dump plan cache information for query](#)  
This Query Supervisor exit program dumps plan cache information about the query that reached a threshold.
- [Exit program to log information using a data queue](#)  
This Query Supervisor exit program shows how to use an asynchronous job to log information about the query that reported a threshold.

# Exit program to send message to QSYSOPR

```

SQS_SNDMSG.CLLE X
SUPERVISOR > QCLLESRC > SQS_SNDMSG.CLLE
57 /*****
58 /*****          Execute          *****/
59 /*****
60
61 /* INIT RETURN CODE TO CONTINUE RUNNING QUERY */
62
63 | | | | CHGVAR   VAR(&RC) VALUE(0)
64
65 /* GET VALUES FROM THE INPUT FORMAT */
66 | | | | CHGVAR   VAR(&JOBNAME) VALUE(%SST(&QRYSD100 13 10))
67 | | | | CHGVAR   VAR(&JOBUSER) VALUE(%SST(&QRYSD100 23 10))
68 | | | | CHGVAR   VAR(&JOBNBR) VALUE(%SST(&QRYSD100 33 6))
69 | | | | CHGVAR   VAR(&THRESHTYPE) VALUE(%SST(&QRYSD100 135 30))
70 | | | | CHGVAR   VAR(&BINHIGH) VALUE(%BINARY(&QRYSD100 165 4))
71 | | | | CHGVAR   VAR(&BINLOW) VALUE(%BINARY(&QRYSD100 169 4))
72 | | | | CHGVAR   VAR(&THRESHVAL) VALUE(%DEC(&BINHIGH 15 0) * +
73 | | | | | | | | &MAXINT + %DEC(&BINLOW 15 0))
74
75 /* GENERATE MESSAGE TEXT STRING */
76 | | | | CHGVAR   VAR(&MSGTXT) VALUE('QUERY SUPERVISOR +
77 | | | | | | | | THRESHOLD TYPE ' *BCAT &THRESHTYPE *TCAT +
78 | | | | | | | | ', THRESHOLD VALUE ' *BCAT
79 | | | | | | | | %CHAR(&THRESHVAL) *BCAT 'REACHED IN JOB ' +
80 | | | | | | | | *BCAT &JOBNBR *TCAT '/' *TCAT &JOBUSER +
81 | | | | | | | | *TCAT '/' *TCAT &JOBNAME )
82 | | | | CHGVAR   VAR(&MSGLEN) VALUE(%LEN(&MSGTXT))
83 | | | | CHGVAR   VAR(%BINARY(&MSGDTA 1 2)) VALUE(&MSGLEN)
84 | | | | CHGVAR   VAR(&MSGDTA) VALUE(&MSGDTA *TCAT &MSGTXT)
85 | | | | SNDPGMSG  MSGID(SQL7064) MSGF(QSQLMSG) MSGDTA(&MSGDTA) +
86 | | | | | | | | TOMSGQ(*SYSOPR)
87
88 /*****
89 /*****          Normal end of program          *****/
90 /*****

```

# Exit program to end query

```

SQS_ENDQRY.SQLRPGLE ×
SUPERVISOR > QRPGLSRC > SQS_ENDQRY.SQLRPGLE > SQS_endqry
55
56 dcl-proc SQS_endqry; // Main procedure
57   dcl-pi *n;
58   | input likes(QQQ_QRYSV_QRYS0100_t);
59   | rc int(10);
60   end-pi;
61
62 //*****
63 // Init the return code to continue running the query
64 //*****
65   rc = 0;
66
67 /* Check Threshold type against SQL Query Supervisor Thresholds Table */
68
69 EXEC SQL
70   | SELECT THRESHOLD_VALUE into :sqs_threshold_value FROM SUPERVISOR.SQS_TRSHLD
71   | WHERE THRESHOLD_NAME = :input.Threshold_Name and ACTIVE = '*YES';
72
73 If SQLSTATE = '00000';
74   //note 00000 = no errors or warning
75   if input.Threshold_Consumption_Value >= sqs_threshold_value;
76   | rc = 1; /* Terminate the query */
77   //   02000 = no data
78   // <do something?>
79   endif;
80   endif;
81   return;
82 end-proc;
```

# Exit program to dump plan cache information for query

```

SQS_DMPQRY.CLLE X
SUPERVISOR > QCLLESRC > SQS_DMPQRY.CLLE
52  /*****/
54  /* INIT RETURN CODE TO CONTINUE RUNNING QUERY */
55
56  | | | | CHGVAR    VAR(&RC) VALUE(0)
57
58  /* GET VALUES FROM THE INPUT FORMAT */
59
60  | | | | CHGVAR    VAR(&BINHIGHU) VALUE(%BINARY(&QRY50100 67 4))
61  | | | | CHGVAR    VAR(&BINLOW) VALUE(%BINARY(&QRY50100 71 4))
62  | | | | CHGVAR    VAR(&PLANID) VALUE(%DEC(&BINHIGHU 15 0) * +
63  | | | | | | | | &MAXINT + %DEC(&BINLOW 15 0))
64
65  /* SUBMIT JOB TO DUMP THE PLAN CACHE FOR THIS QUERY */
66  | | | | SBMJOB    CMD(RUNSQL SQL('CALL QSYS2.DUMP_PLAN_CACHE(' +
67  | | | | | | | | *BCAT 'FILESHEMA => ''SUPERVISOR'', +
68  | | | | | | | | FILENAME => ''PLANDUMPS'', +
69  | | | | | | | | PLAN_IDENTIFIER=> '' *TCAT +
70  | | | | | | | | %CHAR(&PLANID) *TCAT ''')) +
71  | | | | | | | | JOBQ(QUSRNOMAX)
72
73  /*****/
74  /*****          Normal end of program          *****/
75  /*****/

```

# SQL procedure used to log information using a data queue

```

DECLARE V_QUERY_IDENTIFIER VARCHAR ( 8 ) CCSID 1208 ,
DECLARE V_QUERY_PLAN_IDENTIFIER DECIMAL ( 20 , 0 ) ;
DECLARE CONTINUE HANDLER FOR SQLEXCEPTION
BEGIN
GET DIAGNOSTICS CONDITION 1
LOCAL_SQLCODE = DB2_RETURNED_SQLCODE , LOCAL_SQLSTATE = RETURNED_SQLSTATE ,
V_MESSAGE_TEXT = MESSAGE_TEXT ;
CALL SYSTOOLS . LPRINTF ( 'process_SQS_data_queue() - failed with SQLCODE:' CONCAT
ERROR_SQLCODE CONCAT ' SQLSTATE:' CONCAT ERROR_SQLSTATE CONCAT ' and MESSAGE:'
CONCAT V_MESSAGE_TEXT ) ;
END ; /* end of continue handler */
WHILE ( 1 = 1 ) DO
SELECT MESSAGE_DATA_BINARY
INTO V_MESSAGE_DATA_BINARY
FROM
TABLE (
QSYS2 . RECEIVE_DATA_QUEUE (
DATA_QUEUE => 'SQS_DQ' , DATA_QUEUE_LIBRARY => 'SUPERVISOR' ,
WAIT_TIME => - 1 ) -- any negative means to wait forever for the next message
) ;

-- Convert the UTF16 data to UTF8
SET V_MESSAGE_DATA_BINARY_LENGTH = LENGTH ( V_MESSAGE_DATA_BINARY ) / 2 ;
SET V_MESSAGE_DATA_UTF8 = (
SELECT
INTERPRET (
VARBINARY_FORMAT ( HEX ( V_MESSAGE_DATA_BINARY_LENGTH ) ) CONCAT V_MESSAGE_DATA_BINARY
AS DBCLOB ( 63 K ) CCSID 1200 )
FROM SYSIBM . SYSDDUMMY1 ) ;

```

# SQL procedure used to log information using a data queue

```
CREATE TABLE SUPERVISOR.SUPERVISOR_LOG FOR SYSTEM NAME SUPERLOG (
  THRESHOLD_TIMESTAMP FOR COLUMN WHENHIT  TIMESTAMP DEFAULT NULL ,
  JOB_NAME VARCHAR(10) CCSID 37 DEFAULT NULL ,
  JOB_USER VARCHAR(10) CCSID 37 DEFAULT NULL ,
  JOB_NUMBER FOR COLUMN JOBNUM  VARCHAR(6) CCSID 37 DEFAULT NULL ,
  SUBSYSTEM VARCHAR(10) CCSID 37 DEFAULT NULL ,
  USER_NAME VARCHAR(10) CCSID 37 DEFAULT NULL ,
  THRESHOLD_NAME FOR COLUMN THRESHNAME VARCHAR(30) CCSID 1208 DEFAULT NULL ,
  THRESHOLD_TYPE FOR COLUMN THRESHTYPE VARCHAR(30) CCSID 1208 DEFAULT NULL ,
  THRESHOLD_CONSUMPTION_VALUE FOR COLUMN THRESHVAL BIGINT DEFAULT NULL ,
  OPERATION_TYPE FOR COLUMN OP  SMALLINT DEFAULT NULL ,
  SQL_STATEMENT_TEXT FOR COLUMN STMTTEXT  VARCHAR(10000) CCSID 1208 DEFAULT NULL ,
  HOST_VARIABLE_LIST FOR COLUMN HVLIST  VARCHAR(10000) CCSID 1208 DEFAULT NULL ,
  CLIENT_ACCTNG FOR COLUMN "ACCTNG" VARCHAR(255) CCSID 1208 DEFAULT NULL ,
  CLIENT_APPLNAME FOR COLUMN "APPLNAME" VARCHAR(255) CCSID 1208 DEFAULT NULL ,
  CLIENT_PROGRAMID FOR COLUMN "PROGRAMID" VARCHAR(255) CCSID 1208 DEFAULT NULL ,
  CLIENT_USERID FOR COLUMN "USERID" VARCHAR(255) CCSID 1208 DEFAULT NULL ,
  CLIENT_WRKSTNNAME FOR COLUMN "WRKSTNNAME" VARCHAR(255) CCSID 1208 DEFAULT NULL ,
  QUERY_IDENTIFIER FOR COLUMN QRO_HASH  VARCHAR(8) CCSID 1208 DEFAULT NULL ,
  QUERY_PLAN_IDENTIFIER FOR COLUMN PLAN_ID  DECIMAL(20, 0) DEFAULT NULL )

RCDFMT SUPERLOG ;

LABEL ON TABLE SUPERVISOR.SUPERVISOR_LOG
  IS 'SQL Query Supervisor Log Table' ;

GRANT ALTER , DELETE , INDEX , INSERT , REFERENCES , SELECT , UPDATE
ON SUPERVISOR.SUPERVISOR_LOG TO RUDI WITH GRANT OPTION ;
```

```
-- Run once only
CL:CRDTAQ DTAQ(SUPERVISOR/SQS_DQ) MAXLEN(45000) FORCE(*YES) SIZE(*MAX2GB 1000)
  AUTORCL(*YES) TEXT('SQL Query Supervisor Threshold Processor');
```

```
-- Make sure this job is submitted after every IPL
CL:SBMJOB CMD(RUNSQL SQL('call supervisor.process_SQS_data_queue()')) COMMIT(*NONE)) JOB(SQS_MGR)
JOBQ(QUSRNOMAX) INLSPGRP(*NONE) LOG(4 00 *SECLVL) JOBMSGQFL(*PRTWRAP) CCSID(37);
stop;
```

# Bring it all together

```
SELECT *  
  FROM QSYS2.QUERY_SUPERVISOR;  
Stop;  
SELECT *  
  FROM supervisor.SQL_QUERY_THRESHOLD for update ;  
stop;
```

```
CALL  
supervisor.SQL_QUERY_SUPERVISOR_MANAGER (  
  "ACTION" => '*STR');
```

# Bring it all together

```
-- Review Query Supervisor Exit Point Program
```

```
SELECT *
```

```
  FROM qsys2.EXIT_PROGRAM_INFO
```

```
  WHERE exit_point_name LIKE 'QIBM_QQQ_QRY_SUPER';
```

```
stop;
```

```
-- Unregister Exit Point Programs
```

```
CL:RMVEXITPGM EXITPNT(QIBM_QQQ_QRY_SUPER) FORMAT(QRYS0100) PGMNBR(*ALL);
```

```
stop;
```

# Bring it all together

-- Register Exit Point Programs

```
CL:ADDEXITPGM EXITPNT(QIBM_QQQ_QRY_SUPER) FORMAT(QRYS0100)
      PGMNBR(*LOW) PGM(supervisor/SQS_SNDTQ) THDSAFE(*YES)
      TEXT('Query Supervisor send data to *DTAQ');
```

-- Activation Command => SQL Query Supervisor - Send Message to QSYSOPR

```
CL:ADDEXITPGM EXITPNT(QIBM_QQQ_QRY_SUPER) FORMAT(QRYS0100)
      PGMNBR(*LOW) PGM(supervisor/SQS_SNDMSG) THDSAFE(*YES)
      TEXT('Query Supervisor send message to QSYSOPR');
```

-- Activation Command => SQL Query Supervisor - Dump Query

```
CL:ADDEXITPGM EXITPNT(QIBM_QQQ_QRY_SUPER) FORMAT(QRYS0100)
      PGMNBR(*LOW) PGM(supervisor/SQS_DMPQRY) THDSAFE(*YES)
      TEXT('Query supervisor dump plan cache');
```

-- Activation Command => SQL Query Supervisor - End Query

```
CL:ADDEXITPGM EXITPNT(QIBM_QQQ_QRY_SUPER) FORMAT(QRYS0100)
      PGMNBR(*LOW) PGM(supervisor/SQS_ENDQRY) THDSAFE(*YES)
      TEXT('Query Supervisor End Query');
```

stop;

# Bring it all together

```
-- testing SQL statement
SELECT qsys2.qcmdexc('DLYJOB 5'),
       HIST.*
FROM TABLE (
         QSYS2.HISTORY_LOG_INFO(START_TIME => CURRENT TIMESTAMP - 1 HOUR)
       ) HIST
FETCH FIRST 150 ROWS ONLY;
```

# Q & A

Thank you