

User Manual

for

SQL Observer

- a subset of .



Global iSeries Application Performance Analyzer

V06M01E



www.giapa.com

Table of contents:

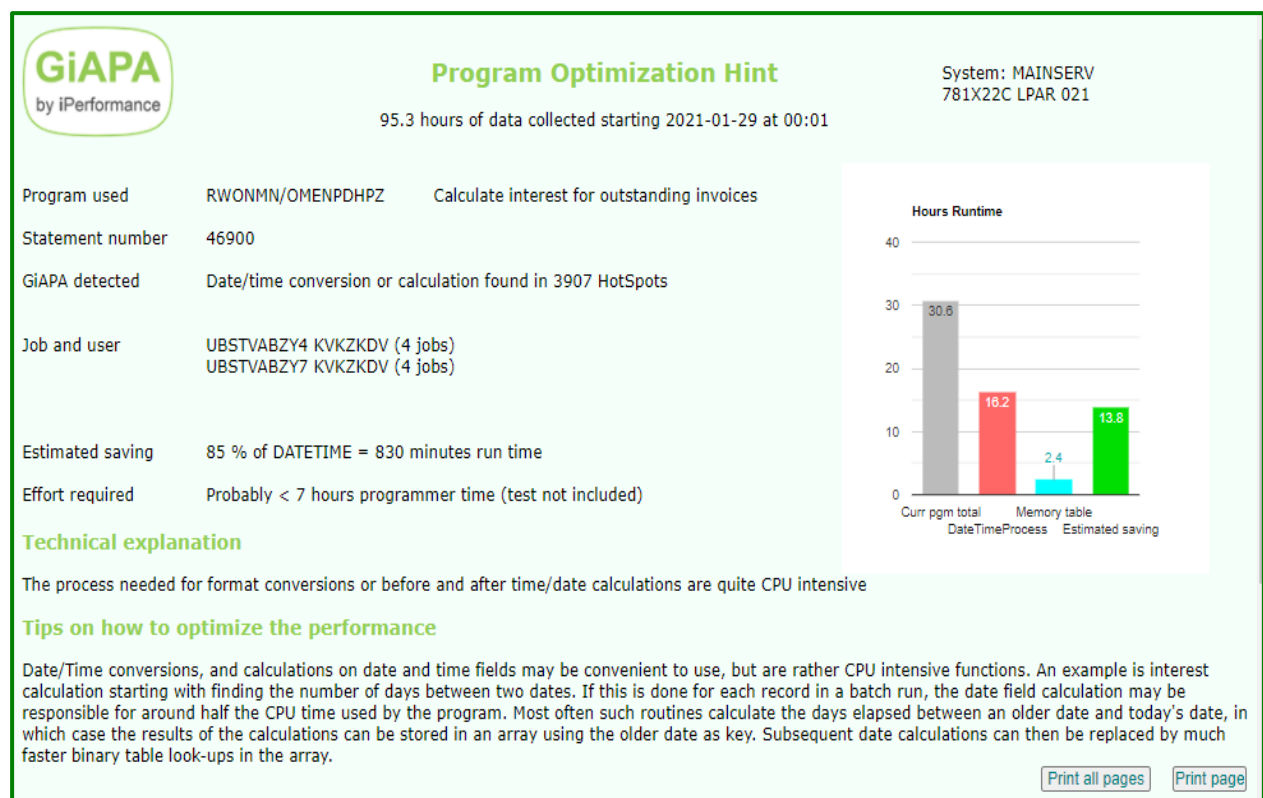
Page 2:	Brief introduction to the software product GiAPA
Page 3:	Introduction to GiAPA’s “SQL Observer” feature
Page 4:	Submit job to collect Plan Cache dumps for SQL Statements
Page 5:	Explanation of how the data collection works
Page 6:	Displaying the “SQL Observer” results
Page 7:	Displaying Current User names How to stop “SQL Observer” data collection
Page 8:	Installation of the “SQL Observer” and the software security code Uninstalling “SQL Observer”

SQL Observer is a subset of the software product GiAPA™

Brief introduction to GiAPA: The objective of GiAPA is to enable the average programmer and operator to identify and solve performance inefficiencies in applications running on the IBM Power servers under IBM i. GiAPA does not compete with IBM performance tools - it offers something else, amongst others an AI-inspired analysis of program function efficiency.

Launched in 2003 and continuously being updated and improved, GiAPA’s over 100.000 lines of source code comprise a software product having very many features. The most advanced is the

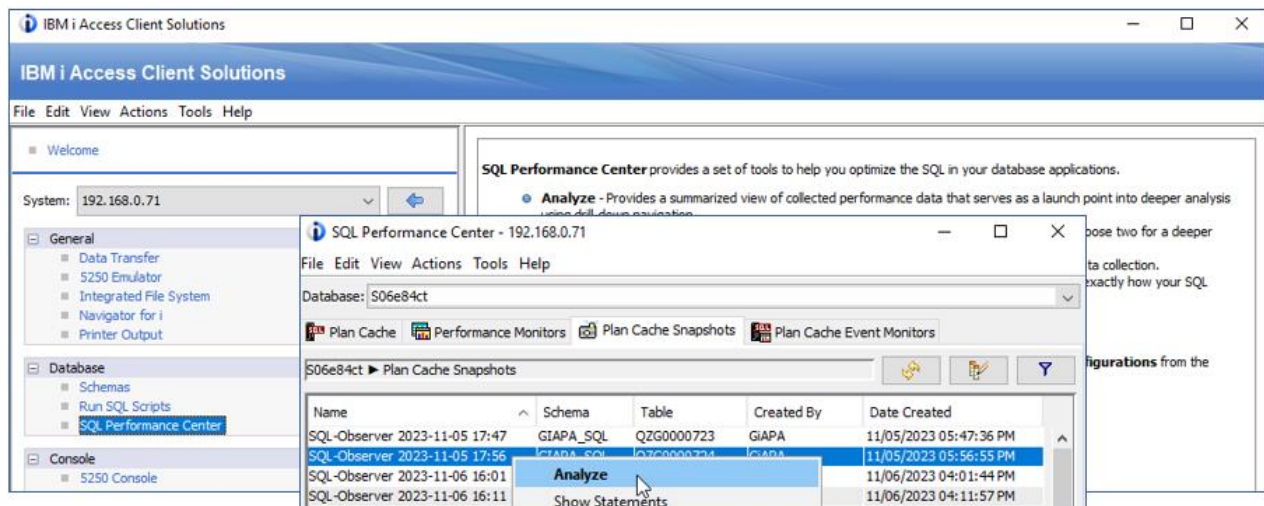
Fully automatic performance analysis of all jobs running on an LPAR shown below, documenting inefficiencies and solutions. The GiAPA data collection uses less than 0.1 % CPU.



Complete GiAPA introduction, references, downloads, etc.: www.giapa.com

SQL Observer: Plan Cache dumps based on Job Watcher data

The intensive and increased use of SQL has made optimization of frequently used SQL statements one of the most rewarding ways of saving server resources. IBM’s “SQL Performance Center” in the Database section of the Access Client Solution (ACS) tool is the gold standard for analysis of Access Plan snapshots from the Plan Cache. Therefore, analysis of the efficiency of SQL statements was not part of the initial design of GiAPA.



Collected Plan Cache data available within ACS.

The Plan Cache data documenting the access methods selected by the Query Optimizer is maintained dynamically in main storage when SQL is running. If the run environment changes, e.g. due to other jobs running, the Access Plan for an SQL statement may be changed. This results in generation of new Plan Cache data, often causing a notable change of the run time.

However, the Plan Cache data is not dumped automatically given it would consume excessive resources. An IBM performance expert recently suggested that a tool offering an automated and user-controlled dumping of Access Plans that might be wanted for analysis could be a popular option. This idea is implemented as GiAPA’s “SQL Observer” available on the GiAPA Menu.

The unique QRO code identifying an SQL activity (one or more SQL-statements accessing a table) must be supplied when requesting a dump of Access Method information. Therefore, the first step for GiAPA’s SQL Observer is to run IBM’s Job Watcher, requesting the QRO code(s) for job(s) specified by the user. At the same time the user defines the frequency for returning Job Watcher data and for dumping Access Plans. An additional parameter defines the number of days the collected data is kept.

SQL Observer shows

- ✓ Job identification and SQL statement(s)
- ✓ Hexadecimal QRO code assigned
- ✓ Reason(s) for selecting Access Plan
- ✓ Snapshot time and Library/file name
- ✓ Changes of Access Plan

This provides numerous possibilities: a special situation may justify collection of data every few seconds e.g. for one or a few jobs. This results in very detailed information collected for these selected case(s) without overall using excessive resources for the data collection. For the normal everyday workload, collection of data every two or five minutes may suffice.

One of the columns available within the collected Job Watcher data contains the Current User Name, which often is wanted in connection with analyzing heavy resource usage. GiAPA’s SQL Observer also includes displaying user names per job and collection interval.

Command GIAPA610: Submit Job GIAPAJWCOL (Data Collection)

```

Submit JW SQL data collection (GIAPA610)

Type choices, press Enter.

Data library name . . . . . DATALIB      GIAPALIB      Name
Data collect. duration minutes  RUNMINUTES  *NOMAX      5-1435, *NOMAX, *NONE
JW collection interval seconds  JWCOLSECS   60          3-3600
Plan Cache dump interv.minutes  PCDMPMINUT  15          3-30
Days to keep Plan Cache dumps   KEEPPCDAYS  7           5-999
Job name . . . . . JOB                  _____ Name, generic*, *ALL
User name . . . . . _____          Name, generic*, *ALL
Job number . . . . . _____          000000-999999, *ALL
                                + for more values
                                _____
                                _____

Additional Parameters

Job queue for submit of job . . JOBQ      QSYSNOMAX    Name
Library . . . . . _____          QSYS        Name, *LIBL

```

DATALIB defines the name of the library where the collected Plan Cache data is stored.

RUNMINUTES defines how long time the data collection should run. If *NOMAX is used for RUNMINUTES, the collection will continue until stopped by using command GIAPA630. This command can also be initiated from GiAPA Menu option 63.

Specify *NONE to only remove data older than KEEPPCDAYS – no collection will be started.

JWCOLSECS defines the interval in seconds between each collection of Job Watcher data. This obviously also affects the resources used by the collection – very frequent collections imply somewhat higher CPU usage and data volume.

PCDMPMINUT defines how often the Plan Cache dumps is scheduled. The Job Watcher data collection routine will be interrupted shortly, allowing GiAPA to dump the Plan Cache data for the QRO codes collected.

KEEPPCDAYS defines how long time the Plan Cache data fetched by GiAPA is stored. GiAPA will automatically delete expired collections.

JOB may be used for selection of max 20 (generic) job names, user names, and/or job numbers, thereby excluding all other jobs from this SQL Access Plan data collection.

Please refer to the job log in case of any errors – keyword parameters from this command are used to generate an ADDJWDFN command – IBM’s rules for that command apply also here.

JOBQ has as default QSYSNOMAX defined within subsystem QSYSWRK – this is the queue normally used for, e.g. performance data collection.

An SQL activity can be based on one or more SQL statements, together defining the SQL function. GiAPA does not save more than five statements per QRO code, and displays only the first three codes which normally is sufficient to identify the case.

A prerequisite for correct interpretation of the results is understanding how the data collection works. The illustration below provides a quick overview.

A unique 4-bytes hexadecimal QRO code identifies an SQL activity (= one or more statements) which within a certain job accesses a given set of data consisting of one or more file members. The QRO codes are needed to request plan cache dumps of the access plans used.

Every JWCOLSECS (Job Watcher Collection Seconds) interval, Job Watcher saves for the selected jobs

1. the name, user, and number for active jobs using SQL,
2. the SQL statement(s) “in progress” or “last completed” for each job, and
3. the unique QRO code identifying each SQL activity.

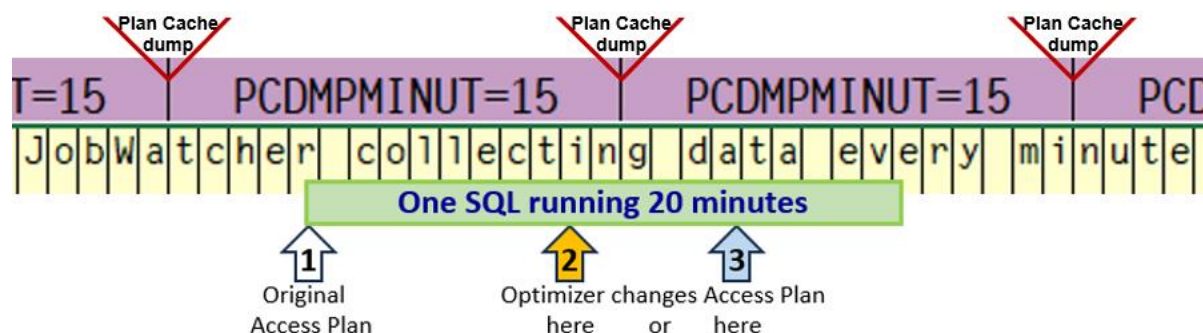
SQL Observer’s default value of 60 seconds between each JW collection will normally suffice. Only jobs starting and ending within one interval are not included. To analyze SQLs within longer running jobs only, a larger interval value can save resources while still supplying sufficient data.

The JW data collection is interrupted at the end of every PCDMPMINUT (Plan Cache Dump Minutes) interval, where control is returned to GiAPA’s SQL Observer to allow dumping of plan cache data for the QRO codes collected by JW during that PCDMPMINUT interval. The dumped access plans are needed as input for the IBM ACS SQL Performance Center if further analysis is needed.

An access plan has a unique plan number and consists of several 10K+ rows/records containing many columns/fields. Although plan cache dumping does not use much CPU, frequent collections for many jobs may obviously occupy much space. Each access plan is only dumped once per PCDMPMINUT interval.

The below example has JWCOLSECS set at 60 (Job Watcher data saved once per minute), and uses PCDMPMINUT=15 thereby passing data to the SQL Observer every 15 minutes.

The green square below illustrates a job running an SQL statement for 20 minutes. The narrow yellow fields represent one minute each, illustrating the JWCOLSECS=60 intervals. Job Watcher stores QRO code(s) representing active SQL(s) at the end of each 60 seconds interval. SQL Observer receives this data at the end of each PCDMPMINUT interval, and requests access plan dumping for each QRO code.



If re-optimization takes place at arrow number 2 (orange arrow), both access plans will be dumped at the end of the first PCDMPMINUT interval where the job is active, and only the new access plan is dumped at the end of the next PCDMPMINUT interval.

But if the re-optimization takes place at arrow number 3 (blue arrow), the first plan cache dump will only include the original access plan, and both plans will be dumped at the end of the following interval.

Command GiAPA62 – Display Collected Plan Cache Data

```

Show Collected Plan Cache Data (GIAPA620)

Type choices, press Enter.

Data library name . . . . . DATALIB      GIAPALIB      Name
Only multiple Access Plan QROs  ONLYMULTIP *NO          *Yes, *NO
Select (generic) job name . . . JOBNAME     *ALL          Name, generic*, *ALL
Select (generic) user name . . . USERNAME   *ALL          Name, generic*, *ALL
Select date/time YYMMDDhhmmss . . STARTTIME 010101000000 Character value
Select date/time YYMMDDhhmmss . . ENDTIME   991231235959 Character value
  
```

DATALIB defines the library containing the dumped Plan Cache data and the GiAPA tables used to control the display of the results.

ONLYMULTIP allows selection of only re-optimized QROs (more than one Access Plan saved).

JOBNAME and **USERNAME** allow (generic) selection of the jobs to be shown.

STARTTIME and **ENDTIME** allow defining time limits for the data to be shown.

GiAPA (c) by iPerformance Plan Cache Snapshots of SQL Access Plan Data 24-03-22 09:54:58

Selections specified: Job: TSTJOIN* Start date/time: 24-03-21 00:00
User: *ALL End date/time: 99-12-31 23:59

Job Name	User Name, JobNbr	Run Date	QRO(Hex)	Nbr of SQL stmts	SQL-Statement Library/SourceFile(Member)
TSTJOIN01	KAARE 126523	2024-03-21	A8D77AD7	2	SQL-stmt(s) from GIAPA_SQL/QRPGLESRC(TSTSQLJOIR)

213 bytes total length
42 bytes: FETCH CURSOR1 INTO : H , : H , : H , : H

171 bytes: DECLARE CURSOR1 CURSOR FOR SELECT LNNAME , CSJNAM , CSJSTA , CSTSTA FROM GIAPALIB . GIAPA143P5 , GIAPALIB . GIAPA143P2
WHERE GIAPA143P5 . LNRRN = GIAPA143P2 . CSACTPCKEY

Dumps available for this plan, last 3 are shown

Text explaining Plan Cache "Access Plan Reason Code"

PlanNbr	Table or member recreated.
2	Table Scan
1	AcPlan Rebuilt
1	Optim.Timeout
1	Generic Info
1	Tmp.HashTabCrt

Plan Cache record types, generated by Query Optimizer when "considering" access plan to select

Alternative Access Plan(s) recorded for this QRO

PlanNbr	Access plan was built to use a reusable Open Data Path (ODP) and optimizer chose a non-reusable ODP for this call
1	Index Used
3	Index Created
2	Temp. Table
1	Table Locked
1	AcPlan Rebuilt
1	Array HostVar
1	Generic Info
3	Distin.Process
2	Grouping
1	Recurs.TabExpr

1 Dumps 2024-03-21 00:18 GIAPA_SQL/QZG0001448

PlanNbr 32551 None of the 25 defined specific reasons for choice of access method apply in this case.

PlanNbr	Table or member recreated.
2	Table Scan
1	AcPlan Rebuilt
1	Optim.Timeout
1	Generic Info
1	Tmp.HashTabCrt

Please observe that the results shown here only are random examples of texts that may appear.

Enter=Go to top F2=Cmd Line F3=Exit F6=Show Current Users PageUp/PageDown

The pages are displayed in ascending sequence by Job Id and QRO code.

Each page contains the data belonging to one QRO code (= SQL activity) within a job. One job may have accessed many different SQL statements, each resulting in a displayed page. Data from a maximum of three different Access Plans are shown.

The SQL statement(s) belonging to the QRO code are shown in green. A QRO can cover several SQL statements, but rarely more than three, which is the maximum displayed.

The documentation of the Plan Cache dumps collected includes the date, time, and names of the latest three files containing Plan Cache snapshots. This information, together with the QRO code, is necessary to locate the corresponding data within the IBM ACS SQL Performance Center, when a performance analysis is required.

If an Access Plan is re-optimized, data for a maximum of two additional plans are displayed which normally is sufficient. If more re-optimizations are expected, any remaining may be seen by using the STARTTIME and ENDTIME keywords to limit the time frame.

When different jobs run the same SQL statement(s) against the same set of data, the same QRO code(s) are saved repeatedly, leading to identical results shown for several jobs.

F6= Show Current User from the above panel displays the following panel with four columns of current users and the date and time where the users were attached to the job.

GiAPA (c) by iPerformance		Current User Names for Job TSTJOIN04 KAARE 102603		23-12-15 16:46:59	
Date and Time	Current User	Date and Time	Current User	Date and Time	Current User
23-11-28 12:52:10	CASASALEX	23-11-28 12:48:30	DCCCADMIN	23-11-28 12:44:49	CASASALEX
23-11-28 12:52:00	ALSLOGJDBC	23-11-28 12:48:20	DCCCADMIN	23-11-28 12:44:39	DCCCADMIN
23-11-28 12:51:50	CASASALEX	23-11-28 12:48:10	CASASALEX	23-11-28 12:44:29	CASASALEX
23-11-28 12:51:40	DCCCADMIN	23-11-28 12:48:00	ROBOKADM	23-11-28 12:44:19	CASASALEX
23-11-28 12:51:30	CASASALEX	23-11-28 12:47:50	CASASALEX	23-11-28 12:44:09	ALSLOGJDBC
23-11-28 12:48:50	DCCCADMIN	23-11-28 12:45:09	ALSLOGJDBC	23-11-28 12:41:28	DCCCADMIN
23-11-28 12:48:40	DCCCADMIN	23-11-28 12:44:59	ROBOKADM	23-11-28 12:41:18	DCCCADMIN
				23-11-28 12:37:38	CASASALEX

Enter=Go to top F2=Cmd Line F3=Return PageUp/PageDown

Additional details are available in these &DATALIB files, easy to access by SQL or Query:

GIAPA612P1	Dumped Plan Cache Snapshot per job and QRO code
GIAPA612P2	SQL-statements and their assigned QRO code
GIAPA612P5	Current user names from Job Watcher QAPYJWTDE

Command GiAPA630 – Stop SQL Observer collection

Stop GiAPA's SQL-Observer (GIAPA630)

Type choices, press Enter.

Stop JW SQLdata collection? . . TERMINATE N Y, N

Use of command GIAPA630 TERMINATE(Y) will cause an active SQL Observer collection of Plan Cache data to terminate at the end of the current PCDMPMINUT interval.

Installation of SQL Observer

The software can be downloaded from <https://www.giapa.com/giapasql.zip>. The downloaded file must be unzipped on a PC using e.g. WinZip. The password needed to open the zipped file can be obtained from iPerformance ApS or from a GiAPA distributor.

Remember that if FTP is used to transfer the downloaded save file to the server, you must use FTP command **bin** to run in binary mode, and the receiving save file should be created on the server before you start uploading from the PC.

Installation of SQL Observer is simply done by the running restore command:

```
RSTLIB  SAVLIB(GIAPALIB) DEV(*SAVF) SAVF(your-save-file-name)
```

Authority needed: The SQL Observer data collection must run under a user profile having E = Execute authority to the Job Watcher commands adding and deleting definitions and starting a data collection.

Command GIAPA009: Install GiAPA Software Security Code

A valid software security code must be installed using CL-command GIAPALIB/GIAPA009.

Set GiAPA security code (GIAPA009)	
Type choices, press Enter.	
Software security code SECCODE	<u>XXXXXXXXXXXXXXXXXXXXXXX</u>
Software update code UPDATECODE	<u>99999</u>

SECCODE: The security code must always be specified.

UPDATECODE: The update code is not always used. It will be supplied when needed. If only the security code is supplied, the update code should be left unchanged.

(To **uninstall** SQL Observer simply use CL-command **DLTLIB GIAPALIB**.)