Documentation of the “SQL Observer” feature

V06M00

Subset of the Global iSeries Application Performance Analyzer User Manual

TM

www.giapa.com

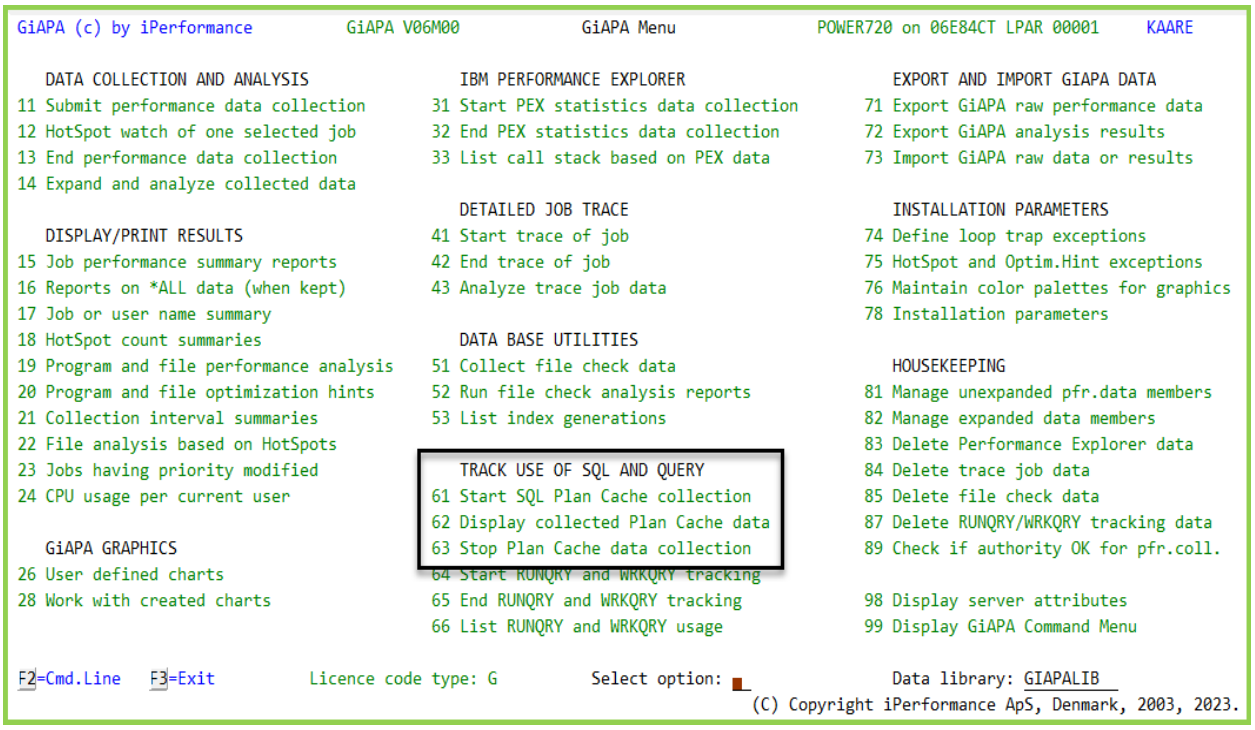
This subset of the GiAPA Manual only documents one feature, the “SQL Observer”, available from options 61 – 63 on the GiAPA Menu.

Table of contents:

Link to additional GiAPA documentation:

https://www.giapa.com/GiAPA_Links.pdf

Use command `GIAPALIB/GIAPA` to display the GiAPA Menu.

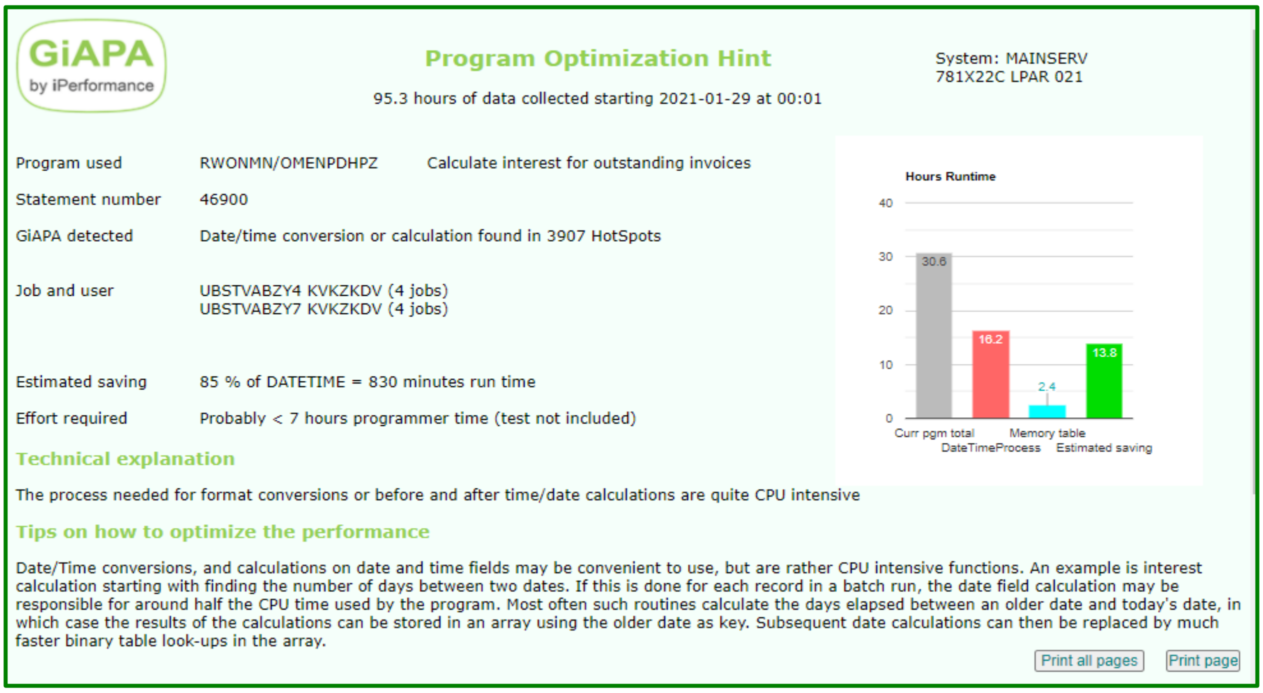


```
GiAPA (c) by iPerformance       GiAPA V06M00        GiAPA Menu          POWER720 on 06E84CT LPAR 00001    KAARE

   DATA COLLECTION AND ANALYSIS        IBM PERFORMANCE EXPLORER              EXPORT AND IMPORT GIAPA DATA
11 Submit performance data collection  31 Start PEX statistics data collection  71 Export GiAPA raw performance data
12 HotSpot watch of one selected job   32 End PEX statistics data collection    72 Export GiAPA analysis results
13 End performance data collection     33 List call stack based on PEX data     73 Import GiAPA raw data or results
14 Expand and analyze collected data
                                          DETAILED JOB TRACE                    INSTALLATION PARAMETERS
   DISPLAY/PRINT RESULTS               41 Start trace of job                  74 Define loop trap exceptions
15 Job performance summary reports     42 End trace of job                    75 HotSpot and Optim.Hint exceptions
16 Reports on *ALL data (when kept)    43 Analyze trace job data              76 Maintain color palettes for graphics
17 Job or user name summary                                                   78 Installation parameters
18 HotSpot count summaries                DATA BASE UTILITIES
19 Program and file performance analysis  51 Collect file check data             HOUSEKEEPING
20 Program and file optimization hints  52 Run file check analysis reports    81 Manage unexpanded pfr.data members
21 Collection interval summaries        53 List index generations             82 Manage expanded data members
22 File analysis based on HotSpots                                            83 Delete Performance Explorer data
23 Jobs having priority modified          TRACK USE OF SQL AND QUERY          84 Delete trace job data
24 CPU usage per current user           61 Start SQL Plan Cache collection    85 Delete file check data
                                        62 Display collected Plan Cache data  87 Delete RUNQRY/WRKQRY tracking data
   GiAPA GRAPHICS                       63 Stop Plan Cache data collection    89 Check if authority OK for pfr.coll.
26 User defined charts                  64 Start RUNQRY and WRKQRY tracking
28 Work with created charts             65 End RUNQRY and WRKQRY tracking     98 Display server attributes
                                        66 List RUNQRY and WRKQRY usage       99 Display GiAPA Command Menu

F2=Cmd.Line   F3=Exit       Licence code type: G       Select option: _        Data library: GIAPALIB
                                                   (C) Copyright iPerformance ApS, Denmark, 2003, 2023.
```

Brief Introduction to GiAPA

The objective of GiAPA is to enable the average programmer, operator, or systems analyst to cope with i.e., identify and solve performance inefficiencies in applications running on the IBM i (Power Systems including AS/400, iSeries, and System i). It was never meant to compete with the various performance tools from IBM – GiAPA works differently and offers something else.

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, which will analyze all jobs running on an LPAR while using less than 0.1 % CPU, and automatically document inefficiencies, including suggested solutions.



Program Optimization Hint

95.3 hours of data collected starting 2021-01-29 at 00:01

System: MAINSERV
781X22C LPAR 021

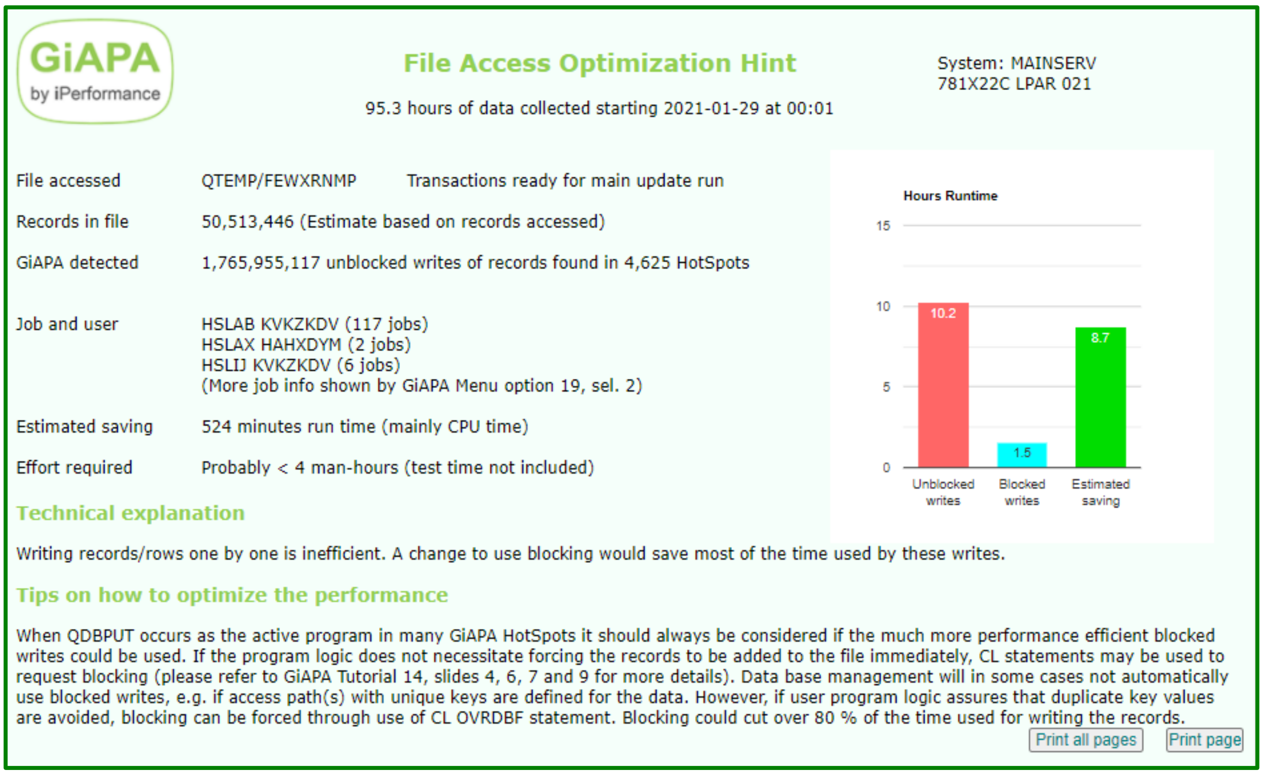
Program used	RWONMN/OMENPDHPZ	Calculate interest for outstanding invoices
Statement number	46900	
GiAPA detected	Date/time conversion or calculation found in 3907 HotSpots	
Job and user	UBSTVABZY4 KVKZKDV (4 jobs) UBSTVABZY7 KVKZKDV (4 jobs)	
Estimated saving	85 % of DATETIME = 830 minutes run time	
Effort required	Probably < 7 hours programmer time (test not included)	

Technical explanation

The process needed for format conversions or before and after time/date calculations are quite CPU intensive

Tips on how to optimize the performance

Date/Time conversions, and calculations on date and time fields may be convenient to use, but are rather CPU intensive functions. An example is interest calculation starting with finding the number of days between two dates. If this is done for each record in a batch run, the date field calculation may be responsible for around half the CPU time used by the program. Most often such routines calculate the days elapsed between an older date and today's date, in which case the results of the calculations can be stored in an array using the older date as key. Subsequent date calculations can then be replaced by much faster binary table look-ups in the array.



File Access Optimization Hint

95.3 hours of data collected starting 2021-01-29 at 00:01

System: MAINSERV
781X22C LPAR 021

File accessed	QTEMP/FEWXRNMP	Transactions ready for main update run
Records in file	50,513,446 (Estimate based on records accessed)	
GiAPA detected	1,765,955,117 unblocked writes of records found in 4,625 HotSpots	
Job and user	HSLAB KVKZKDV (117 jobs) HSLAX HAHXDYM (2 jobs) HSLIJ KVKZKDV (6 jobs) (More job info shown by GiAPA Menu option 19, sel. 2)	
Estimated saving	524 minutes run time (mainly CPU time)	
Effort required	Probably < 4 man-hours (test time not included)	

Technical explanation

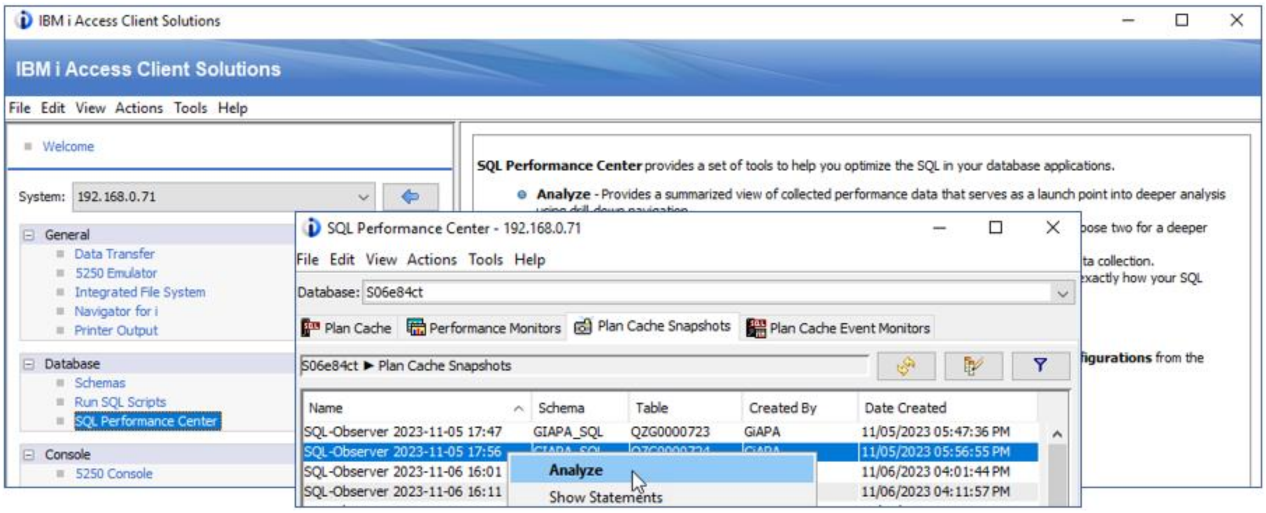
Writing records/rows one by one is inefficient. A change to use blocking would save most of the time used by these writes.

Tips on how to optimize the performance

When QDBPUT occurs as the active program in many GiAPA HotSpots it should always be considered if the much more performance efficient blocked writes could be used. If the program logic does not necessitate forcing the records to be added to the file immediately, CL statements may be used to request blocking (please refer to GiAPA Tutorial 14, slides 4, 6, 7 and 9 for more details). Data base management will in some cases not automatically use blocked writes, e.g. if access path(s) with unique keys are defined for the data. However, if user program logic assures that duplicate key values are avoided, blocking can be forced through use of CL OVRDBF statement. Blocking could cut over 80 % of the time used for writing the records.

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.



SQL Performance Center provides a set of tools to help you optimize the SQL in your database applications.

- **Analyze** - Provides a summarized view of collected performance data that serves as a launch point into deeper analysis

Name	Schema	Table	Created By	Date Created
SQL-Observer 2023-11-05 17:47	GIAPA_SQL	QZG0000723	GiAPA	11/05/2023 05:47:36 PM
SQL-Observer 2023-11-05 17:56	GIAPA_SQL	QZG0000724	GiAPA	11/05/2023 05:56:55 PM
SQL-Observer 2023-11-06 16:01				11/06/2023 04:01:44 PM
SQL-Observer 2023-11-06 16:11				11/06/2023 04:11:57 PM

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.

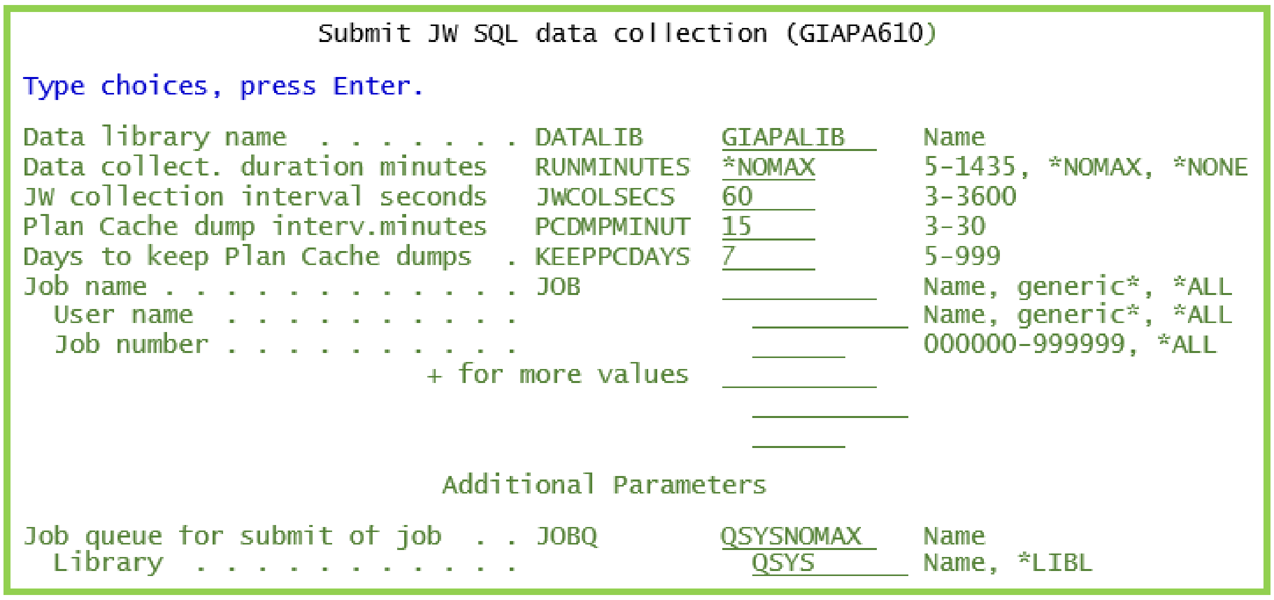
However, the Plan Cache data is not dumped automatically given it would consume excessive resources. An IBM performance expert recently suggested that a tool offering an automated and user-controlled dumping of Access Plans that might be wanted for analysis could be a popular option. This idea is implemented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

The unique QRO code identifying an SQL activity (= one or more statements) 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.

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.

GiAPA Menu Option 61: 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 to understand how the data collection works. The illustration below provides a quick overview.

For every JWCOLSECS interval, Job Watcher will for the selected jobs save the Job-Ids running SQL, the SQL statements, and the QRO codes identifying the SQL statements. The data collected is passed to GiAPA's SQL Observer at the end of every PCDMPMINUT interval.

A given QRO code may be included

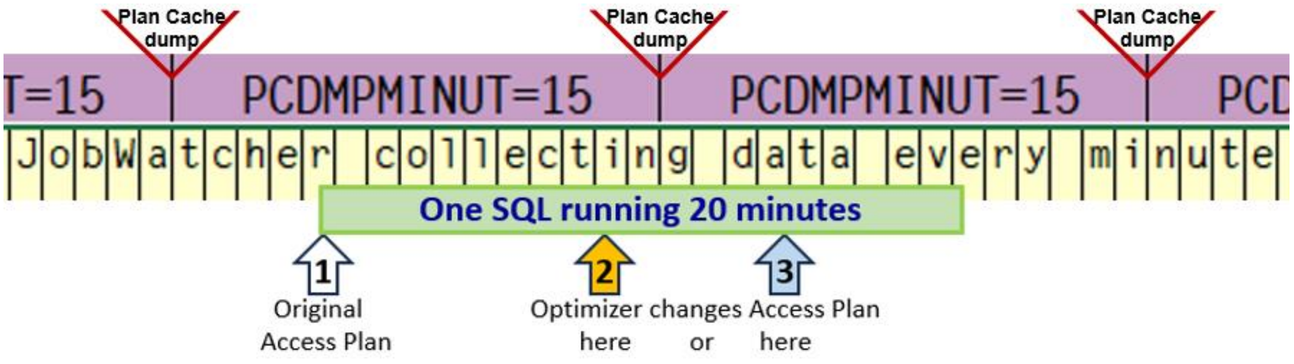
- only once if the SQL statement only was active briefly within only one job, or
- repeatedly if the SQL statement was active longer time and/or within more jobs.

GiAPA's SQL Observer will for each QRO code request a Plan Cache dump of the Access Plan which can serve as input for the IBM ACS SQL Performance Center. A dumped Access Plan consists of several rows/records, each having a length of more than 10K and containing 282 rows/fields. Although the data collection does not use much CPU, the large volume of data forces the user to avoid collecting data for any unnecessary jobs.

Because data only is dumped at the end of a PCDMPMINUT interval, the dump for a QRO code will reflect the Plan Cache data active at that point of time. A re-optimization replaces the old Access Plan, which therefore becomes unavailable.

The below example uses JWCOLSECS set at 60 which saves Job Watcher data once per minute, and PCDMPMINUT set at 15 meaning the data is passed to the SQL observer every 15 minutes.

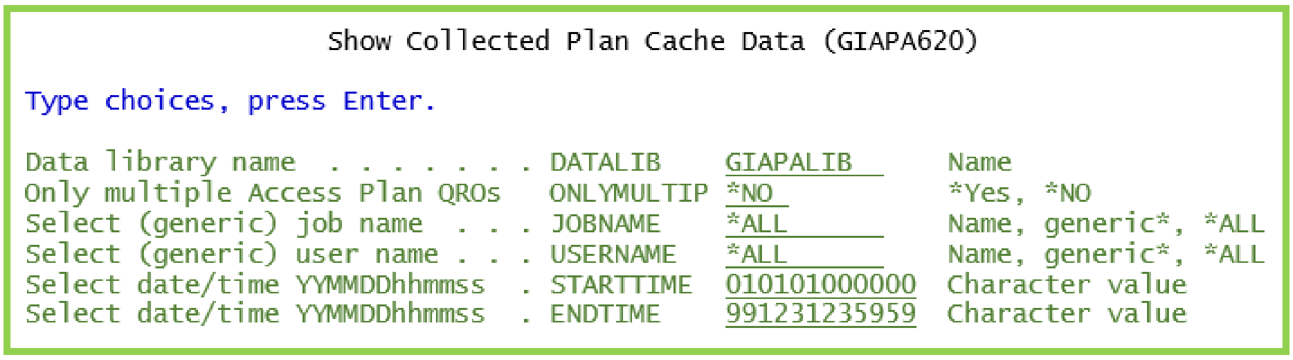
The green square below illustrates a job running the same SQL statement for 20 minutes. The narrow yellow fields represent one minute each, illustrating the JWCOLSECS=60 intervals. Job Watcher stores QRO code(s) representing active SQL(s) of the selected job(s) at the end of each 60 seconds interval.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



If re-optimization takes place at arrow number 2 (orange arrow), the Plan Cache dumps at the end of the two PCDMPMINUT intervals will both contain the new Access Plan, because the original plan was replaced before the end of the first interval.

But if the re-optimization takes place at arrow number 3 (blue arrow), the first Plan Cache dump will reflect the original, not yet replaced Access Plan, and the second dump will contain the new plan. Should re-optimization occur both at arrow 2 and arrow 3, then plans 2 and 3 are dumped.

GiAPA Menu option 62 – Display Collected Plan Cache Data



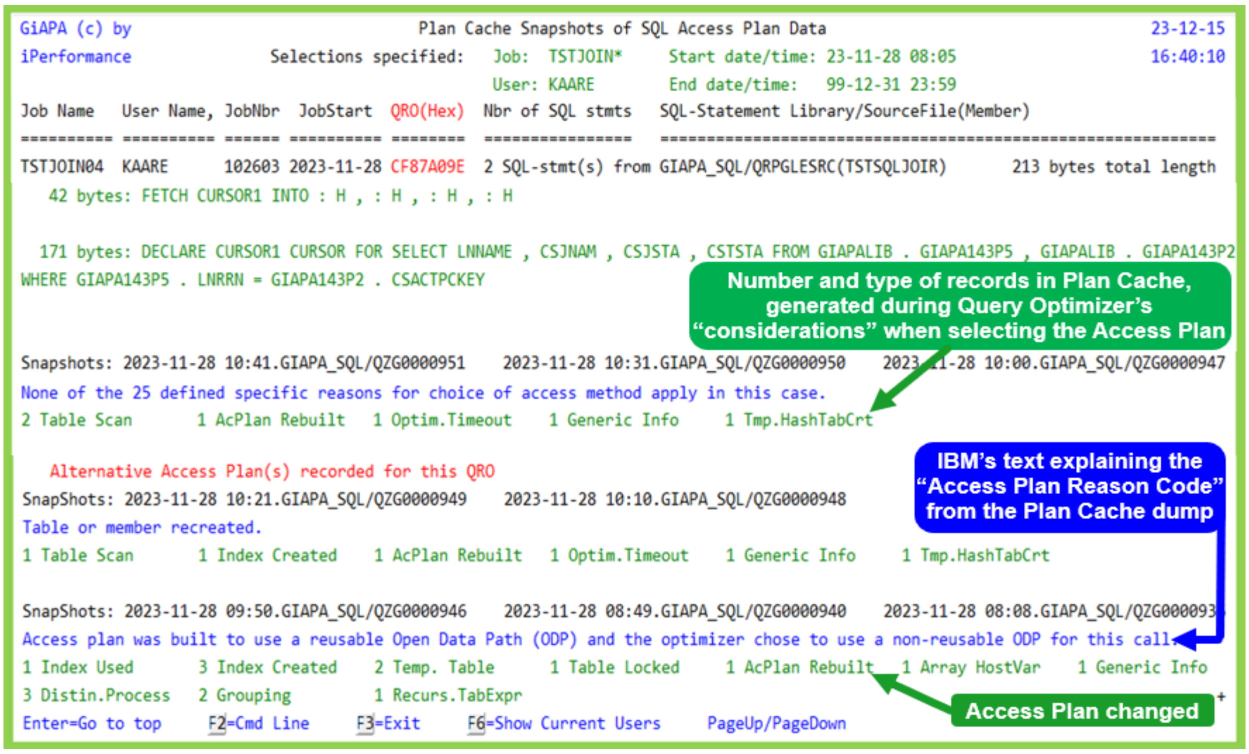
```
                      Show Collected Plan Cache Data (GIAPA620)

 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                        Plan Cache Snapshots of SQL Access Plan Data                          23-12-15
iPerformance             Selections specified:   Job:   TSTJOIN*      Start date/time: 23-11-28 08:05     16:40:10
                                                 User: KAARE          End date/time:   99-12-31 23:59
Job Name   User Name, JobNbr  JobStart  QRO(Hex)  Nbr of SQL stmts   SQL-Statement Library/SourceFile(Member)
========== ========== ====== ========== ========  ================   ==============================================================
TSTJOIN04  KAARE      102603 2023-11-28 CF87A09E  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


Snapshots: 2023-11-28 10:41.GIAPA_SQL/QZG0000951    2023-11-28 10:31.GIAPA_SQL/QZG0000950    2023-11-28 10:00.GIAPA_SQL/QZG0000947
None of the 25 defined specific reasons for choice of access method apply in this case.
2 Table Scan        1 AcPlan Rebuilt   1 Optim.Timeout     1 Generic Info      1 Tmp.HashTabCrt

   Alternative Access Plan(s) recorded for this QRO
SnapShots: 2023-11-28 10:21.GIAPA_SQL/QZG0000949    2023-11-28 10:10.GIAPA_SQL/QZG0000948
Table or member recreated.
1 Table Scan        1 Index Created    1 AcPlan Rebuilt    1 Optim.Timeout     1 Generic Info      1 Tmp.HashTabCrt

SnapShots: 2023-11-28 09:50.GIAPA_SQL/QZG0000946    2023-11-28 08:49.GIAPA_SQL/QZG0000940    2023-11-28 08:08.GIAPA_SQL/QZG000093
Access plan was built to use a reusable Open Data Path (ODP) and the optimizer chose to use 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                                                              +
Enter=Go to top     F2=Cmd Line        F3=Exit       F6=Show Current Users      PageUp/PageDown
```

Number and type of records in Plan Cache, generated during Query Optimizer's "considerations" when selecting the Access Plan
IBM's text explaining the "Access Plan Reason Code" from the Plan Cache dump
Access Plan changed

The Access Plan Reasons and the Plan Cache record types listed above ("Table Scan", etc.) do not apply for the shown SQL statements – they are random examples of texts that may appear.

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.

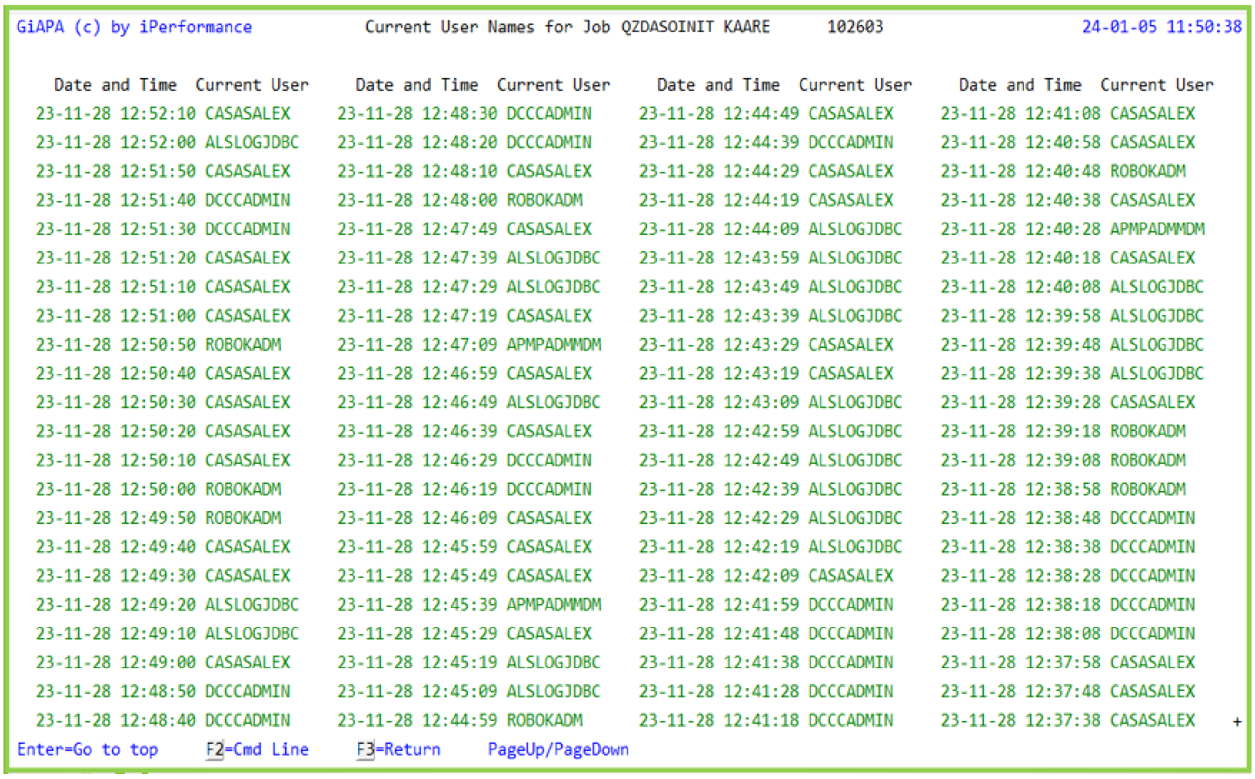
The SQL statement(s) belonging to the QRO code are shown in green. A QRO can cover several SQL statements, but rarely more than three, which is the maximum displayed.

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.

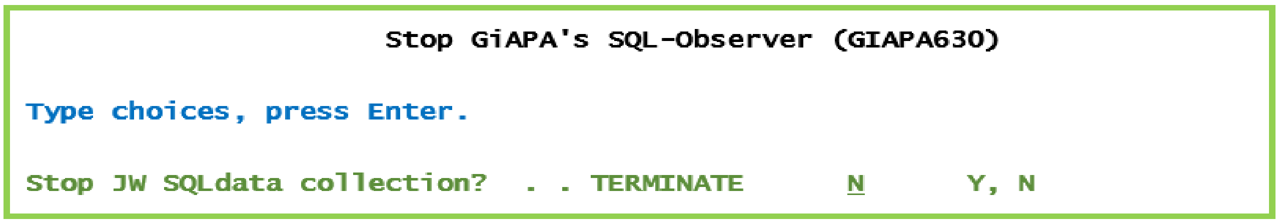
Please note that more than one job may run the same SQL statement(s), thereby causing the same QRO code(s) being saved which in turn leads to identical results shown for several jobs.

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 QZDASOINIT KAARE      102603              24-01-05 11:50:38

    Date and Time  Current User    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 12:41:08 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 12:40:58 CASASALEX
  23-11-28 12:51:50 CASASALEX    23-11-28 12:48:10 CASASALEX    23-11-28 12:44:29 CASASALEX    23-11-28 12:40:48 ROBOKADM
  23-11-28 12:51:40 DCCCADMIN    23-11-28 12:48:00 ROBOKADM     23-11-28 12:44:19 CASASALEX    23-11-28 12:40:38 CASASALEX
  23-11-28 12:51:30 DCCCADMIN    23-11-28 12:47:49 CASASALEX    23-11-28 12:44:09 ALSLOGJDBC   23-11-28 12:40:28 APMPADMMDM
  23-11-28 12:51:20 CASASALEX    23-11-28 12:47:39 ALSLOGJDBC   23-11-28 12:43:59 ALSLOGJDBC   23-11-28 12:40:18 CASASALEX
  23-11-28 12:51:10 CASASALEX    23-11-28 12:47:29 ALSLOGJDBC   23-11-28 12:43:49 ALSLOGJDBC   23-11-28 12:40:08 ALSLOGJDBC
  23-11-28 12:51:00 CASASALEX    23-11-28 12:47:19 CASASALEX    23-11-28 12:43:39 ALSLOGJDBC   23-11-28 12:39:58 ALSLOGJDBC
  23-11-28 12:50:50 ROBOKADM     23-11-28 12:47:09 APMPADMMDM   23-11-28 12:43:29 CASASALEX    23-11-28 12:39:48 ALSLOGJDBC
  23-11-28 12:50:40 CASASALEX    23-11-28 12:46:59 CASASALEX    23-11-28 12:43:19 CASASALEX    23-11-28 12:39:38 ALSLOGJDBC
  23-11-28 12:50:30 CASASALEX    23-11-28 12:46:49 ALSLOGJDBC   23-11-28 12:43:09 ALSLOGJDBC   23-11-28 12:39:28 CASASALEX
  23-11-28 12:50:20 CASASALEX    23-11-28 12:46:39 CASASALEX    23-11-28 12:42:59 ALSLOGJDBC   23-11-28 12:39:18 ROBOKADM
  23-11-28 12:50:10 CASASALEX    23-11-28 12:46:29 DCCCADMIN    23-11-28 12:42:49 ALSLOGJDBC   23-11-28 12:39:08 ROBOKADM
  23-11-28 12:50:00 ROBOKADM     23-11-28 12:46:19 DCCCADMIN    23-11-28 12:42:39 ALSLOGJDBC   23-11-28 12:38:58 ROBOKADM
  23-11-28 12:49:50 ROBOKADM     23-11-28 12:46:09 CASASALEX    23-11-28 12:42:29 ALSLOGJDBC   23-11-28 12:38:48 DCCCADMIN
  23-11-28 12:49:40 CASASALEX    23-11-28 12:45:59 CASASALEX    23-11-28 12:42:19 ALSLOGJDBC   23-11-28 12:38:38 DCCCADMIN
  23-11-28 12:49:30 CASASALEX    23-11-28 12:45:49 CASASALEX    23-11-28 12:42:09 CASASALEX    23-11-28 12:38:28 DCCCADMIN
  23-11-28 12:49:20 ALSLOGJDBC   23-11-28 12:45:39 APMPADMMDM   23-11-28 12:41:59 DCCCADMIN    23-11-28 12:38:18 DCCCADMIN
  23-11-28 12:49:10 ALSLOGJDBC   23-11-28 12:45:29 CASASALEX    23-11-28 12:41:48 DCCCADMIN    23-11-28 12:38:08 DCCCADMIN
  23-11-28 12:49:00 CASASALEX    23-11-28 12:45:19 ALSLOGJDBC   23-11-28 12:41:38 DCCCADMIN    23-11-28 12:37:58 CASASALEX
  23-11-28 12:48:50 DCCCADMIN    23-11-28 12:45:09 ALSLOGJDBC   23-11-28 12:41:28 DCCCADMIN    23-11-28 12:37:48 CASASALEX
  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
```

GiAPA Menu option 63 – 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 GiAPA

The software can be downloaded from https://www.giapa.com/giapa.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 **i**Performance ApS or from a GiAPA distributor.

Authority needed: Some IBM performance collector APIs used by GiAPA are shipped with *PUBLIC authority *EXCLUDE. Therefore, installation or update of GiAPA must be made by a user profile having QSECOFR authority, and with the system value QALWOBJRST allowing a restore of programs using adopted authority. Alternatively, GiAPA data collection must run under a user profile having authority to use these APIs.

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.

GiAPA is installed simply by restoring GIAPALIB. When the unzipped save file containing GIAPALIB has been transferred to the iSeries using e.g. FTP, run the following command:

```
RSTLIB  SAVLIB(GIAPALIB) DEV(*SAVF) SAVF(savefilename)
```

Important: Modification to any automatic backup routine for GIAPALIB

If GIAPALIB is backed up using “save while active” during performance data collection, error message “Cannot allocate object” should be avoided through specifying `OMITOBJ((GIAPALIB/*ALL *USRSPC) (GIAPALIB/*ALL *USRQ) (GIAPALIB/GIAPA115* *USRIDX))`.

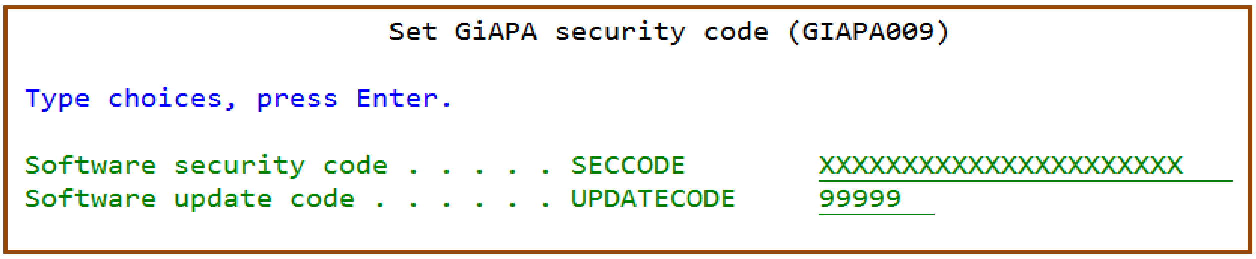
(To **uninstall** GiAPA from the server simply use CL-command **`DLTLIB GIAPALIB`**.)

Command GIAPA009: Install GiAPA Software Security Code

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

Collection of Plan Cache data using GiAPA option 61 does require a valid security code.

GiAPA performance data (Menu option 11) can always be collected and exported – these and a few other functions do not require a valid security code.



```
                    Set GiAPA security code (GIAPA009)

Type choices, press Enter.

Software security code . . . . . SECCODE        XXXXXXXXXXXXXXXXXXXXXX
Software update code . . . . . . UPDATECODE     99999
```

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.