JACKIE's Forum

OS/400 Query Options File

Jackie Jansen

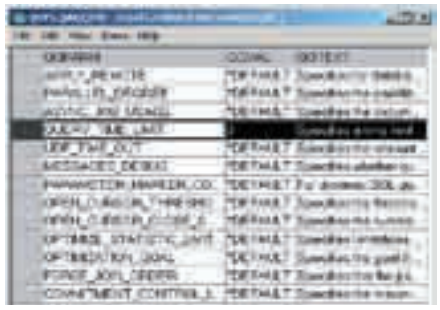
With V4R4, IBM consolidated the various attributes that can affect a querie's performance into one table. No longer do we need to find out if the parameter that we require is in the data area QQQOPTONS, or the system value QQRYDEGREE, or even in CL commands such as CHGQRYA. The query options table QAQQINI has them all. This table can be specified at a system level or an individual job level.

There isn't room in this column to list all the parameters that you can control via the QAQQINI file but here are a few. If you want to set a time limit on a query using the query governor the control is here. If you want to control the parallelism or number of CPUs used in a query you can do that also. Do you want to turn on the query debug messages, optimize your SQL or even use star join queries?

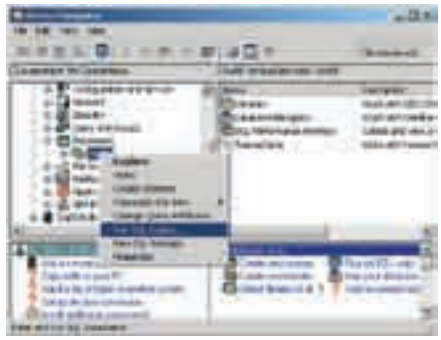
There are some tricks to using the QAQQINI file. First, if you look in QSYS you will find a QAQQINI file. Do not modify this file. It is intended as a template only. To create your own query options file you will want to duplicate the file in QSYS using the Create Duplicate Object (CRTDUPOBJ) command with DATA(*YES) specified. Do not use Copy File (CPYF). This file has associated triggers that will not be duplicated with the CPYF command.

To make your options available to the entire system you will want to create QAQQINI in the QUSRSYS library. To tailor the options to a specific query you should create this file in a user library and specify the library name on the QRYOPTLIB parameter of the CHGQRYA command in the job. If you don't issue a CHGQRYA command or don't specify a QRYOPTLIB command the system will automatically check the QUSRSYS library for a QAQQINI file. Options specified within this file will automatically apply to all jobs.

The QAQQINI file consists of one record for every parameter that can be modified. After duplicating the table from QSYS you will need to modify the appropriate records. For example, to set the query governor to 0 seconds for testing purposes you would find the record where QQPARM equals "QUERY_TIME_LIMIT". You would then modify the QQVAL column to "0". You can make these changes through any data update tool such as SQL or DFU. An easy way to modify the file is to use iSeries Navigator. Open the file in your user library and then directly enter your new parameter values.



Another way to modify this file is to use the IBM Custom Query Options Builder tool on the web. Go to "http://www.ibm.com/iseries/db2/db2tips.htm". You will be asked what library you want your QAQQINI table created in, and what values you want for all the parameters. Each parameter is well documented. After specifying your new parameter's values you will press a "Build" button. At this point a window will pop up with all the SQL script needed to create and modify the QAQQINI table. You can cut and paste this SQL script into the SQL script centre in iSeries Navigator. To do this start iSeries Navigator, right click on either "Databases" or the local system name below Databases and choose "Run SQL Scripts". Using the SQL Script Centre was explained in more detail in the recent "SQL Query Access" article in "Jackie's Forum". Another alternative is to cut and paste the SQL into an iSeries source member. You can then use the OS/400 command "Run SQL Statement (RUNSQLSTM)" to execute the script statements. You do not need to be licensed to the iSeries SQL Development Kit to use this command. OS/400 has full support for runtime SQL embedded in it.



Overall having IBM bring all the different query performance options into one table for access is very handy. You must remember that if you create this table in QUSRSYS and then modify it, the modifications will automatically apply to all jobs running on the system unless you override them. You have the capability here of causing major query performance degradation. For example, if you wanted to specify that one of your queries was a star join query but used the QAQQINI table in QUSRSYS you would successfully decrease the run time of your join query but you would have increased the run time of all your standard queries. Don't forget to use unique user libraries when you want to update any parameter that was not intended to apply to the entire system. TUG

***Jackie Jansen** is a Senior Consulting IT Specialist. She currently works in the IBM Americas Advanced Technical Support Solutions Centre. Jackie is a frequent speaker at iSeries Technical Conferences and Users Group meetings. Contact her at jjansen@ca.ibm.com.*