Comparing Queries: Vector versus PSQL

A White Paper From

For more information, see our web site at
http://www.goldstarsoftware.com

Comparing Queries: Vector versus PSQL

Last Updated: 09/24/2015

Once you have successfully migrated your data from PSQL to Vector, you should perform some of your own functionality and performance tests to compare the results. These tests will give you the confidence to proceed with the Vector environment for all of your reporting needs.

Your first check should be to run some of your more common queries and verify that both databases are returning the same data set. This is going to be your functional test. You want to use the same query tool to extract the data from the query and then compare the results, using either a text comparison tool or some other solution which compares actual database records (especially if the data comes back in a different order). Some of this difficulty can be mitigated if you include ORDER BY clauses in your queries to ensure that the data gets sorted in the same order on both reports.

Your second check should be for performance. Depending on which tool you select and how that tool is actually implemented, it may be difficult to tell the difference in performance. In this paper, we show how to use the SQLExec utility to run a complex query from the command line and to obtain data results and timing numbers back for each step in the process.

Running SQLExec to Obtain Query Results and Timings

The Goldstar Software tool SQLExec can be used to get simple timings on your queries, including both the time it takes to execute the SQL query, as well as the time it takes to return all the data. SQLExec is a 32-bit Windows tool, and will require a 32-bit ODBC driver be installed (and the DSN be created) for the database you intend to access.

1. First, verify that the proper ODBC drivers are installed for each database environment you will be testing.
2. Second, verify that a 32-bit ODBC DSN has been created for each database environment you will be testing. In the samples below, we will use “MyDSN” to indicate the DSN name.
3. Download the SQLExec tool from www.goldstarsoftware.com/tools.asp and unzip the files into an empty folder on your workstation for easiest access.
4. In the same folder, create a text file (with Notepad or another text editor) and paste in the SQL query that you would like to use for testing. Avoid the use of spaces or other special characters in the filename. In this example, we will use “Query.TXT” for this file.
5. Start a Windows Command Prompt and navigate to the folder.
6. Perform a test run of SQLExec first to make sure that you are getting data back properly. The proper command line will look like this:

```
SQLEXEC MyDSN /MQuery.TXT /Uuser /Ppass
```

7. Once you have the query working, you will want to make TWO changes to this command line. First, we will add the **/DT** switch. This will tell SQLExec to display the starting and ending times for each step in the query process, as well as

compute the elapsed time for the SQL query to execute and for the data to come back. The other change is just as important – we are going to redirect the output to a disk file instead of to the screen. For most systems, disk output is really faster than the screen, so this will result in more accurate timings for a real application. The new command line looks like this:

```
SQLEXEC MyDSN /MQuery.TXT /Uuser /Ppass /DT >test1.txt
```

8. You should always run TWO tests to eliminate the effects that data caching may have on the environment, and use the latter timing.

Once you have the result files for your two environments, you can compare the data in each and verify that they both return the exact same data set. Further, you can see the actual runtimes reported by SQLExec. Here is an example result (without the data):

```
12:50:07.925: Building SQL Environment
12:50:07.925: Start Connection
12:50:08.025: Allocate Statement Handle
12:50:08.025: Calling SQLExecDirect
12:50:08.037: Returned from SQLExecDirect
Query Execution Time: 12 ms
12:50:08.037: Starting Data Read
  [Data gets reported here and has been cut out for clarity]
12:50:08.044: Finished Data Read
Data Read Time: 7 ms
12:50:08.044: Free Statement Handle
12:50:08.044: Disconnect from Database
12:50:08.044: Tearing Down SQL Environment
12:50:08.044: Complete!
```

You can easily see that the SQL query execution time was 12 milliseconds. This query was a straightforward two-table join with a restriction on a string column containing a LIKE clause. Further, you can see that the time it took to read all of the data records from the database (to SQLExec) took 7 milliseconds. Now, run the same query again, but specify the other DSN, and generate a result for the second system. You can now directly compare the performance results, too.

Note that many database product license agreements prohibit reporting performance comparisons without written authorization from the vendor. Because of this limitation, we will not report any comparison numbers. However, the infomration presented in this paper can certainly allow you to see for yourself how your various database engines compare in terms of performance.

If you are not comfortable with this process, Goldstar Software can provide a complete line of database services for you. For more information, contact us at 1-708-647-7665 or visit our web site at http://www.goldstarsoftware.com.