Troubleshooting Query Performance Issues and Solutions

When a query is executed, it can sometimes seem to take an unusually long time before you see any results, or the running of the query may "time out" giving you an error message. Most of these situations can be prevented through careful structuring of the query and attention to detail. This includes testing output incrementally, the choice and ordering of tables/records, the logical use of joins, the selection and ordering of criteria, the economical utilization of fields, and the use of expressions.

More information: Here are links to documentation in the SFIS Blog with further tips and details on query performance.

PeopleSoft Query – Advanced Concepts, Lesson 11
Query – Encyclopedia / Glossary
Query Tip Sheet
Query - Expressions
Wesleyan Functions

The following are some typical situations and suggested strategies on how to fine-tune query performance as to response time through an understanding of the foundations of the query process. If you have a query that is performing slowly, timing out or returning an unmanageable number of records, here are some items to look at.

Run Time:

After you initially build a query, the first time it is run it will be slower than on later tries.

- The query may be saved, but is not compiled until the first time it is run.
    a. Behind the scenes, Oracle maps out an execution plan and saves it in memory, so it will usually run faster on subsequent attempts.

Columns / Fields:

Only pull the columns / fields that you need into the query.

- Not all tables are populated in PeopleSoft, and not all fields in all tables are populated.

Restart and Test the Query, Incrementally:

To test the query, create it anew, one step at a time.

- Begin with one table, if feasible, and test some criteria to see if the information returned is reasonable, and once you are satisfied, add the next table and so on.

PS Query and Security:

PS Query passes through row-level security, which restricts whose data you can see.

- If you don't normally have access to view data through the pages/panels, you will be subject to the same restrictions in PS Query.

Joins:

The quality of the joins can have a pronounced impact on the results.

- Preventing a runaway query.
  - Be sure to join all appropriate **key fields** to ensure that the query does not time out.
- Because PeopleSoft is a **relational database**, data are stored in multiple tables.
  - A single page/panel does not necessarily imply that the data shown there are stored in a single table.
- Not all tables can be successfully joined.
  - If at all possible, join on related records.
  - If there are no fields that match up, you will return either no data or a very large amount of meaningless data, a **Cartesian join**, which is a join of every row of one table to every row of another table.
- Related Records are records/tables that have their relationships/joins established on the database side, rather than defined in the query itself.
  - While query does a good job of detecting the appropriate join conditions, it is not perfect, so be sure the joins make sense.
- Adding a description column.
  - Note that asking for Translate Table values or description fields from other tables will add more joins and more overhead.
  - Where possible, use an **Expression** or a **Wes Function** to supply a description.

- Avoiding duplicate rows.
  - To eliminate duplicate rows, you may have to try several methods, for example, joining on more than one field.
  - The existence of many duplicates in the query record set may mean that you performed a Cartesian join.
  - The DISTINCT property setting:
    - In general, avoid using the DISTINCT property setting in PS Query.
    - Except in the case of a very small record set, the DISTINCT property may dramatically decrease the performance of the query.
    - However, depending upon the makeup of the query's elements, DISTINCT may be necessary.

Tables / Records:

Things to consider regarding Tables are the Size, Complexity, Number, and Ordering

- Size / Complexity.
  - If the query eventually completes running, albeit slowly, take a look at the tables used and consider if it makes sense for the process to be slowed because of the size of the underlying tables and the complexity of the request.
  - Consider counting Rows by using the Testing Table Size method to see if reordering makes sense.
- Number.
  - Note that the fewer the tables that you use as part of your query, the better.
- Ordering.
  - The order in which you bring records or views into the PS Query grid is important.
  - PS Query and Oracle have built-in optimizers that make the code more efficient, but in most cases it helps the query run better if tables that are expected to return the fewest rows are **added last** in the FROM clause.
  - It is important to realize that the order in which tables are added to the query affects the joins that are produced.
  - The SQL statements that PS Query builds are evaluated from the bottom up. However, it will not correct a bad (or invalid) join.
- Testing Table Size.
  - This is one method that may aid in improving query response time through selecting tables in the Query page by size. You can test in order to see an

abbreviated set of data by directing the query to return a limited number of rows.

- Selecting tables in order by size will have a big impact on query performance. Query runs using Oracle, a relational database, which executes a query from the bottom up - the smaller the table, the quicker the results.
  - The first factor in determining the size of a table is simple – the fewer the fields, the smaller the table.
  - If a table has a "_TBL" on the end of its name, it is also normally small.
  - By using an Expression, the amount of data stored in a table can be easily determined.
  - If you test using the Expression of **count(*)** one-by-one for one field in each record, you can fine the size of each record.
  - Then once you have chosen the records to utilize, you would create the new query, selecting the tables from largest to smallest, i.e., they will appear on the Query page in that order.
  - To review the full demonstration, see **Lesson 11, Slide 7, "Return a Limited Number of Rows,"** in PS Query Instruction Part 3 – Advanced Concepts in the SFIS Blog.

Views:

Views – which are used like Records - are saved queries that can be referenced as record sets in other queries.

- Views generally end with a suffix of VW as the naming convention.
  - Some Views are delivered with PS and some are created in ITS.
  - Views are available for various data selection and restriction purposes.
  - Because it is basically a saved query, there is a slight performance hit when joining to a View.

Criteria:

Refines the query by specifying conditions that the retrieved data must meet..

- Amount of Criteria.
  - The more you put in for Criteria, the quicker the query will run.
- Criteria Order.
  - You can sort criteria on the Criteria page to improve the performance of a query.
    - If in your original query you added a criterion related to the Record A

after criteria related to Record B, the query needs to process all the rows in Record A before it can reach Record B.

- See the Query Glossary entry of Boolean Operators on the Criteria page and the linked article on grouping criteria.

- Prompts.
  - If you select a Prompt Table Edit Type, the correct record **should** appear by default on the Edit Prompt Properties page. Carefully review the Edit Prompt Properties page:
    - It is important that you verify that the Prompt Table field displays the record that stores the values that you want users to see.
    - Also, be sure that the Edit Type field and the Prompt Table field complement each other:
      - If Edit Type says Prompt Table, there must be one
      - If Edit Type is blank, then the Prompt Table field must have no value.
    - Before you run the query, be sure any prompts you may have removed from the Criteria page have also been removed from the Prompts page.
      - Although removed elsewhere, Prompts are not removed automatically from the Prompts page.
      - Leftover Prompts can cause the query to fail.

- Multiple Criteria Values.
  - Multiple criteria values would be a case where you need a query that pulls, for example, more than a single major, but not all of them.
  - Build your queries incrementally, pulling the criteria one at a time, rerunning the query after each criterion is added.
  - Set temporary criteria to control the output and see if the query is working properly.
  - There are several ways to query for multiple values. These are methods you can try. They are listed in recommended order.
    - The In List operator.
    - Multiple OR statements
    - The UNION operator
    - The Like Operator
    - Subquery
  - Alternatively, you may want to see if Connected Query would be appropriate.