

What Your Mother Never Told You About the Query Window

Janet E. Stuelpner, ASG, Inc., Cary, North Carolina

ABSTRACT

The query window is a point and click way to extract data from a table. With very little instruction, you can create an SQL program to gather data, manipulate it and create reports. Does it have limitations? The answer to that is yes. Is there a work around for those limitations? The answer is also yes. This paper will explore what some of those limitations are in the query window and how to program around them.

INTRODUCTION

The SQL Query Window is an interactive application. It simplifies the process of building and executing an SQL Query. Data that is stored in SAS® data sets, SAS views, VSAM files, Database Management Systems (DBMS) supported by SAS/ACCESS® software, or any data the SAS System can access are accessible through the Query Window. Without being familiar with SQL or the SAS SQL procedure, you can build, save and run queries. You can run a report of your query, save your query as a catalog query entry to include later, save your query as a catalog source entry, save your query to an external file, or save your query results as an SQL table (SAS data set) or SQL view. In order to build a query, you must choose what actions you want to be performed on the data. The difference here is that you are doing it all in a point and click environment and then have the ability to save the code for changes that you would like to make in the future. As with any procedure you can customize a report by applying SAS formats and labels to individual columns, build new columns, set specific WHERE conditions to subset your data, or request distinct values for your variables. You can summarize, group, and order the data as with any SAS procedure. From the actions you choose from the drop down program menus (PMENUS), the Query Window creates an SQL query. The SQL query can be composed of many components; however, only a table and its columns are required to build a complete query.

Some of the workarounds you will find here are done completely within the query window itself. By manipulating the constants you can type in what ever is necessary to get your code running. However, the answer in some cases is to save the code that is generated by the query window and change or

modify it in the program editor. After this is done, the code can then be run. In either case, generation of the source code in the query window can save a great deal of time and effort.

SIMPLE QUERIES

There are several ways in which to invoke the query window. In the SAS Program Editor, you can type in the Query command. You can type it on the command line and hit enter or type it in the command box and click on the check. If you prefer the total point and click method, you can click on the GLOBAL pmenu, go down to ACCESS and then click on Query. This will open an SQL Query Tables window. The tables that are available from which you can choose are defined by the LIBNAME statements that were submitted in your SAS session. From the available tables list, you can highlight a table, click on the appropriate arrow and move it to the select tables window and click on OK at the bottom left hand corner of the window. This will open the SQL Query Columns window. From here you can choose which columns you want to include in your query by highlighting them and moving over into the selected columns window. By clicking on the LOCALS pmenu you can choose to specify a WHERE clause to subset your data, order the data by choosing a sort variable or grouping the data. There are ways to format the column with a new format or build new columns to include in your query. After you have finished configuring the query, it is time to run the query. This is simply done with PMENUS. Go to ACTION. You can run the query by choosing RUN QUERY. To run the query immediately, then choose RUN IMMEDIATE. However, if you would like, you can review the query before you run it by choosing SHOW QUERY. This will bring up a window with all of the code written out. There are buttons on the side of the window that then allows you to run the query or save it in various forms.

After the query is run, there are several choices you will have to make. You can save the query in many ways: as a query to include later, as a source entry or in an external file. You can also save the results. You can save them as a table or you can create a view.

There are many choices to make even with a simple query. The PMENUS will direct you. If the end

result is a report, the PMENUS will direct you into the REPORT window so that you can create a report from the data. This is another way to point and click your way to creating a report and saving both the report and the code. It is similar in its actions to the query window, but the end result is the report.

COMPLEX QUERIES

My definition of a complex query is not the same that you will encounter in the documentation. The documentation defines a complex query as one with many sub-queries. However, the focus of this paper is to introduce the query window and some of its limitations. These limitations occur when accessing data in a Database Management System that is supporting by SAS/ACCESS software. This includes DB2®, Oracle®, Sybase® and Microsoft Access®. This is where you need to do some creative programming. The thing to remember is that we are accessing data from an alternate data format other than SAS in the SQL Query Window. There are a great number of options that you can employ when accessing the data through PROC ACCESS or the SQL Pass-through facility. However, when using the Query Window, your options are limited because the code is being written for you in a very specific way. Let's discuss some of the problems and the ways to work around them.

For the remainder of the paper, the discussion will focus on two aspects of the query window where problems arise. When working with functions and table joins. The query window has particular idiosyncrasies of which we need to be aware.

Functions

When a query is defined using a database like DB2 or MS Access, there is an inner select statement and an outer select statement. The inner select statement sends instructions to the DBMS itself. In this case, you can use SQL statements and functions that are specific to the DBMS. The outer select sends instructions to SAS. Therefore, with one you can use any available SAS functions, with the other, you can only use the functions that are available to that database management system. For example, there is an INTCK function in SAS that is used to calculate intervals between SAS dates. When used in the inner select with another DBMS, an error condition occurs. However, in both SAS and DB2, there is a DAY, MONTH and YEAR function. When trying to use a SAS function in another DBMS, it causes a problem within the Query Window. The Query Window assumes that you are using SAS functions on a SAS table. So, if your DBMS defines the data type differently than the SAS

system defines it, you cannot use a SAS function. Is there a way to get around this? Of course there is. Let's suppose that the DBMS that you are accessing defines dates as character. We all know that a SAS date is defined as numeric. Therefore, when trying to set up a WHERE clause, SAS assumes that it is numeric in the Query Window. This will cause a problem, because every time you click on a value the quotes that are needed for the character value of a variable will be missing. How do we get around that? The answer lies in the use of the choice of a constant value in the WHERE EXPRESSION window. In the SQL QUERY COLUMNS window, choose the LOCALS PMENU and then click on the "Where Conditions for Subset" This will open the Where Expression window. To create your WHERE clause, click on the <CONSTANT enter values> choice in the available columns window. This opens a constant window where you can type anything you want. In essence, this is the beginning of getting around the restrictions imposed by the query window. If the column in the DBMS is defined as character, choose the CHAR button, otherwise choose the NUM button. Then you can be true to the column type from the DBMS. When done typing in the function you need, choose OK. The query window will guide you to choose an operator and then another value. Again, you can choose CONSTANT and type in the information that will be compatible with the DBMS you are accessing. When you run the query, everything will work correctly based on the fact that you chose the data types in the correct way.

Table Joins

When working in the query window, the first rule of thumb when joining tables is to include all of the tables you want to join when entering the query window. When the list presents itself, it is necessary to choose all of the tables at that time. This takes a bit of planning when preparing to create a query. Unfortunately, SAS has a problem defining the alias for each table if you don't choose the right tables at the beginning of the query. Obviously, if you don't choose all the tables you need, you can save the code and start again.

The second thing of which you need to be aware when joining tables is that a problem exists when the ON clause in an inner, left right and full join. To make it even more mysterious, it only happens when there are multiple conditions in the WHERE clause when the operator is an AND. For some reason, the query window incorrectly sets up the join with an improper alias when two or more WHERE conditions are needed. Is there a workaround for this, of course. Unfortunately, the solution lies outside the query window. The first thing to do is create the

WHERE statement in the manner you believe it needs to be completed. Then, the query must be saved as an external file that can be brought into the program editor. Once you open the file into the program editor, you can modify the code by inserting the correct aliases.

CONCLUSION

Although many people believe that the Query window is a simple tool to be used by people who are beginning programmers, this is a misconception. The strength of the query window is that all of the setup can be done and saved and access of tables is simple and quick. The SAS/ACCESS software requires a descriptor section for each procedure that is run. With a greater knowledge of the way that the Query window works, it can make programming easier. The value in using the query window is that the setup, query and results can be saved into catalogues or external files.

REFERENCES

SAS Institute Inc. (1995) *SAS Guide to the SQL Query Window, Usage & Reference, Version 6, First Edition*, Cary, NC: SAS Institute Inc.

SAS Institute Inc. (1994) *Getting Started with SAS/Access Software, Version 6, First Edition*, Cary, NC: SAS Institute Inc.

SAS Institute Inc. (1989) *SAS Guide to the SQL Procedure, Usage & Reference, Version 6, First Edition*, Cary, NC: SAS Institute Inc.

SAS is a registered trademark or trademark of SAS Institute Inc. in the USA and other countries.

Other brand and product names are registered trademarks of their respective companies

® indicates USA registration.

ACKNOWLEDGMENTS

I would like to take this opportunity to acknowledge all of the help and support given to me by my husband, Bob Stuelpner. He diligently read and reread this paper in an effort to correct the obvious errors and keep me on the right track. His criticisms were constructive and his support never ending.

I would also like to express my appreciation for the help and support given to me by Caroline Bahler of ASG, Inc.

CONTACT INFORMATION

Janet Stuelpner
ASG, Inc.
1100 Summer Street
2nd floor
Stamford, CT 06905

(203) 356-9540 (voice)
(203) 363-3746 (fax)
jes@asg-inc.com