“Bridging the Gap between SAS® Applications Developers and Database Administrators”

Suzanne D. McCoy
Independent Consultant

ABSTRACT
As a consultant, I have been exposed to many different companies, operating platforms, and uses of SAS Software. Every company I have worked with has used some type of relational database for centralized storage of corporate data, and in every case, there has been the high level of misunderstanding between “SAS people” and “database people”.

This paper focuses on the causes of misunderstanding. It also contains tips and techniques for developing a strong working relationship between SAS Applications Developers and Database Administrators.

INTRODUCTION
Most Database Administrators (DBAs) see SAS as a black box. If they have had any exposure at all, they typically have only seen Proc Access. Most sites have the DBA group or a Data Manager extract data into flat files that are then read into SAS data sets. The concept of live data reads has been around for a few years but since “things” work the way they are, many sites have not developed the skills needed to implement live data access. Since most SAS programmers are not prepared to explain the “ins and outs” of SAS Access software, confusion and misunderstanding usually start here, the earliest point in the process. Tension mounts further each time more interaction occurs between the groups. Why? The answer, in my opinion, is the lack of use of a common language and lack of recognition of common goals.

CONCEPTS
Proc Access is a wonderful tool for many uses, however, Proc SQL, with the SQL Pass-through facility, is common ground for both groups. Developed per ANSI SQL standards, Proc SQL allows SAS programmers to use basically the same language as the DBA. If you show a DBA your code for a SAS view built using the SQL Pass-through facility, he can determine exactly what the program will accomplish.

As a SAS programmer, you need to understand that data stored in the database is ultimately and fully the responsibility of the DBA. The DBA has issues and concerns that have to be addressed before anyone is allowed to access the data using a SQL based language. Typical DBA questions include [they meaning the SQL users and ‘we’ meaning the DBA group]:
• Do they understand the data structure, especially of the tables are fully normalized?
• Could they do update the data?
  • Should multi-table views or read-only views be developed for their use to ensure that data updates are not possible?
  • Does table security need to be upgraded since SAS relies solely on table security (versus application specific security measures that may be in place for data input areas)?
• Will we be able to handle the additional load on the database (i.e., will significant performance degradation occur due to live data access)?
• Who will validate the selection criteria and what coding standards will be applied? How will the SAS programmers avoid generating Cartesian products (Notice the terminology here!) since there won’t be any type of query optimizer in use? Can/Should we get included in the code review loop?
• What happens if we need to change the structure of the tables? How can we assess the impact of the change on their code?
• Will we be given a chance to build database views for them if that will improve the efficiency of their extraction?

What typically happens when the DBA starts asking these types of questions? The less knowledgeable SAS programmers get intimidated. Don’t let this happen!

An Applications Developer should welcome these questions because it opens the door for discussions and planning about how to best access the data. Show that you respect the DBA’s knowledge, understand their responsibility for data integrity, and want their input during development of your SAS code. The DBA group may actually write the queries for you so that all you have to do is put the Proc SQL pass-through syntax around them! Just
remember that no matter how fancy the database query language is, SAS only understands the syntax documented in SAS user guides and the database SQL code only understands its syntax. Be sure to get both sets of books out!

Keep in mind that the code contained in the SQL Pass-through step is database SQL not SAS SQL, and every database SQL language is not the same. Some relational databases do not provide for outer unions while others do not allow views to be created using union statements. In these cases, you may need to build individual views using the SQL Pass-through facility and then combine them using SAS SQL. Handy tips to help you relate SAS SQL syntax to SAS data step processing are included in SAS Guide to the SQL Procedure: Usage and Reference, Version 6, First Edition. Two tips to remember are:

- An outer union is like combining data using a set statement.
- An outer union corresponding is like combining data using a merge statement.

ADDITIONAL TIPS

- Work together to maximize efficiency. Ask the DBA to monitor database performance during data extractions and provide feedback.
- Find out if there are coding standards implemented for database SQL queries. If so, use the same standards in SAS SQL Pass-through code.
- Use environment variables for pointing to different databases. This way you can easily switch between production and test environments without having to change any code.
- Utilize a central storage area for SAS database views and the code used to generate them. This way everyone can easily locate and utilize them. When changes to the database tables are being considered, the DBA will be able to assess the impact on the SAS views. You may also find that other users who need the same data can use your views instead of creating new, potentially less efficient ones.
- Unless you have a need to 'freeze' the data, work from views instead of SAS data sets. Once the data is extracted into a data set, the DBA can no longer be responsible for its integrity or security! There may be occasions when it is better for the data to be frozen in the database itself instead of in a SAS data set (e.g., Clinical trials data that needs to be cut-off at a certain date before submission to the FDA). If you do extract the data into SAS data sets, be sure you understand the difference between creation date and last modified date (see Proc Contents and your environment manual for details).

- Find out if there are data dictionaries stored as tables in the database. Most sites using normalized data structures maintain 'code lists' instead of storing character data. Build views of these 'dictionary' tables and use them as input for your SAS formats.
- Use the SAS data dictionaries to 'data-drive' your data extraction code if your database views are standardized. This, when combined with the SAS format (database code lists) tip, eliminates many/most code maintenance issues.
- Don't be afraid to say "I don't know" or "I don't understand your terminology" until you and the DBA gain experience working together on code development.
- Once the communication pathways are established, show the DBA how to use Proc SQL to view SAS data sets or views. This should be reasonably easy for them and will allow them flexibility that may be missing in their database SQL language. As their comfort level with SAS increases, showing them simple procedures like Proc Print, Proc Means (multiple simultaneous summaries) and Proc Freq (item counts - multiple, simultaneous) can make a big difference in the way that they present their data to others. The icing on the cake will be showing them Proc Tabulate since no SQL language does this type of processing without creating numerous temporary tables.

CONCLUSION

The key to successful interaction with the Database Administrator is, and always will be, you! Never forget that both groups have common goals concerning data integrity and efficient processing. Your communication skills, your knowledge of the SAS SQL language, and a little bit of patience are the critical factors to 'bridging the gap'.

REFERENCES


SAS is a registered trademark of SAS Institute Inc. in the USA and other countries. © Indicates USA registration.
Other brand and product names are registered trademarks or trademarks of their respective companies.

ANSI is the accepted abbreviation for the American National Standards Institute.

The Author may be contacted as follows:

Suzanne D. McCoy
PO Box 731
Carolina Beach, NC 28428
or e-mail: 76744.2140@compuserve.com