PharmaSUG 2023 - Paper PO-080

What PROC SQL Can't Handle While Data Step Can?

Deming Li, Merck & Co., Inc., North Wales, PA 19454

ABSTRACT

We've all learned and used Base SAS® and PROC SQL but might not have time to compare the two. This paper gathers around some facts from the author's experiences and puts together some practical findings to guide beginners and hopefully refresh memories for the advanced SAS programmers.

INTRODUCTION

Our codes often consist of data steps or PROC SQL or combination of both, i.e., they are used interchangeably. We don't usually stop and question why. Much has to do with our programming style, background and knowledge. A person who has learned Base SAS® from the very beginning finds the data step is the way to go while the other programmer uses the PROC SQL instinctively as he or she might have switched from a database career to SAS. Reasons are abundant but they don't matter. However, a further look into Base SAS® and PROC SQL themselves reveals the fact that PROC SQL performs better than Base SAS® in some arears while at the same time, it has other shortcomings. This insight is very helpful, at least conceptually, in our daily programming work.

PROC SQL vs SAS® program

It is true that PROC SQL does a lot of things that data step does: it queries the database and retrieves data; creates tables, views and indices; creates new variables as well as macros variables; adds or modifies records; performs sort and join etc.

Both tools work well in terms of data manipulation but there are a lot of differences.

First of all, the terms used are different: dataset vs table; variable vs column; observation vs row; merge vs join etc.

Secondly, the syntax is profoundly different as PROC SQL uses the common SQL language shown below:

```
proc sql ;
  create table adsl_new as
    select a.*
        from adsl as a
          where a.armcd=1 ;
quit ;
```

PROC SQL is basically a SAS version of SQL language that works well in the SAS environment. It bears the nature of the database tool as it works more efficiently at retrieving data thru querying. It can also summarize the data in one run without other procedures like PROC FREQ and/or PROC MEANS. Last but not the least, it can output the result without initiating PROC PRINT.

All the above differences are reflected by the fact that data step in Base SAS® is a procedural programming language, i.e., it does line by line or sequential processing while PROC SQL conducts set or block processing. Hence it is not hard to find that PROC SQL handles the query and summarizing data much more e1fficiently than Base SAS®, especially when handling a large amount of data. Yet on the

other hand, the data step in Base SAS® is much more efficient in manipulating data, especially handling complex business rules using loops etc.

Here are some of the things that PROC SQL can't handle:

1) Base SAS® can work with raw data file like text file or excel but PROC SQL can only work with data already imported in as SAS datasets or in tables stored in a database.
2) Base SAS® has its unique data _null_ step to assign data driven macro variables; conduct report writing etc.
3) Base SAS® transposes data easily by using PROC TRANSPOSE or array in data step while PROC SQL requires additional codes.
4) Base SAS® can do various kinds of arrays or loops within a data step or in a macro but the PROC SQL does not do it well. It uses case-when structure, which is cumbersome to program.
5) Within a macro call, we can use macro variables to pick and choose which line of code in a data step to be executed. It is not so easy for PROC SQL to handle. A lot of times, it has to create more tables than necessary to accomplish that.
6) Base SAS® lets us to use first dot and last dot in situations where we want to keep the first record in a group or the last. Again, PROC SQL does not seem to be able to operate easily.
7) Base SAS® provides data step options like end= and obs= while PROC SQL lacks.
8) Base SAS® also provides statements like retain and lag, that are completely mission impossible for PROC SQL.
9) Base SAS® has PROC SORT with nodupkey that allows us to select distinct observations by the specified by vars. PROC SQL can't get it done this way. It uses the distinct statement which works like the PROC SORT nodup.
10) Base SAS® creates multiple datasets when doing the merge shown below in one run:

```
proc sort data=adsl ;  by subjid ;  run ;
proc sort data=adae ;  by subjid ;  run ;

data adsl_only adae_only both ;
  merge adsl (in=ina)
        adae (in=inb) ;
  by subjid ;
  if ina and inb then output both ;
  else if ina and not inb then output adsl_only ;
  else if inb and not ina then output adae_only ;
run ;
```

But PROC SQL has to create one table at a time and does not give out control when doing such joins i.e. you have to trust it 100% as you don't know if the join is done correctly, especially it does Cartesian join. Chances are you might get more records than expected.

There are a few other things that PROC SQL can't handle during join:

- When datasets get merged in Base SAS®, it is not necessary to list out all the variables. PROC SQL, however, requires us to spell out all the variables intended from the second table or subsequential tables to be joined upon.
- Base SAS® provides a good control during the merge by using the in= option

- Base SAS® allows us to apply complex business rules during the merge in one data step: if then else.

However, PROC SQL has an upper hand over Base SAS® in terms of joining tables: it does not require the tables to be sorted; the join variable names don't have to be the same in the join on or where statements. Furthermore, the Cartesian join performs well if there is a need for many-to-many join, which the BASE SAS® lacks.

CONCLUSION

Data Step or PROC SQL or Both? It is a good question. When knowing the pros and cons of PROC SQL, it is not hard to address it. This paper is certainly not the comprehensive list of them all but hopefully it can be used as a conceptual guide. In conclusion, it is ideal and desirable to obtain both skills and be able to use them where one outperforms the other. At least we know why in some cases the data step is used in place of PROC SQL and vice versa.

REFERENCES

Shankar, Charu, "Why choose between SAS Data Step and PROC SQL when you can have both?", PharmaSUG 2018 Paper QT-09

https://www.pharmasug.org/proceedings/2018/QT/PharmaSUG-2018-QT09.pdf

Craig Dickstein, Ray Pass, "DATA Step vs. PROC SQL: What's a neophyte to do?", SUGI 29 Paper 269-29

https://support.sas.com/resources/papers/proceedings/proceedings/sugi29/269-29.pdf

ACKNOWLEDGEMENT

The author is grateful for Wenyu Hu for her encouragement and the time spent for proofreading this paper.

CONTACT INFORMATION

Your comments or questions are encouraged. Please feel free to contact the author at:

Deming Li

Merck & Co., Inc.

351 N Sumneytown Pike

North Wales, PA 19454

e-mail: deming.li@merck.com