Paper 11741-2016

Secrets of Efficient SAS® Coding Techniques

Andrew T. Kuligowski, HSN
Swati Agarwal

ABSTRACT

Just as there are many ways to solve any problem in any facet of life, most SAS® programming problems have more than one potential solution. Each solution has tradeoffs; a complex program may execute very quickly but prove challenging to maintain, while a program designed for ease of use may require more resources for development, execution, and maintenance. Too often, it seems like those tasked to produce the results are advised on delivery date and estimated development time in advance, but are given no guidelines as to efficiency expectations.

The purpose of this paper is to provide ways for improving the efficiency of your SAS programs. It suggests coding techniques, provides guidelines for their use, and shows the results of experimentation to compare various coding techniques, with examples of acceptable and improved ways to accomplish the same task.

KEYWORDS

Base SAS, Code tips, Macro variables, procedures, conditional code

INTRODUCTION – WHAT IS "EFFICIENCY"

For the purpose of this discussion, efficiency will be defined simply as obtaining more results from fewer computer resources. When you submit a SAS program, the computer must:

- load the required software into memory
- compile the program
- find the data on which the program will execute
- perform the operations requested in the program
- report the result of the operations for your inspection

All of these tasks require time and space. Time and space for a computer program are composed of CPU time, I/O time, memory, and external storage space. Reviewing definitions of these concepts:

- **CPU time:** the time the Central Processing Unit spends performing the specified operations.
- **I/O time:** the time the computer spends on input and output. Input refers to moving the data from storage areas such as disks or tapes into memory while output refers to moving the results out of memory to storage or to a display device.
- **Memory:** the size of the work area that the CPU must devote to the operations in the program.
- **External:** storage space would include things like disk space, CDs, USB sticks, or any input or output external to the CPU and memory.

THE THEORIES AND THEIR PROOF (OR DISPROOF)

The experiments to compare various coding techniques and their results in terms of efficiency or lack thereof were still in progress at the time the Proceedings were scheduled to go to press. As such, the assumptions and proposed trials are described below; their actual results will be reviewed in person at SAS Global Forum in April 2016.

CODING TIP #1:

When performing calculations make sure you execute only necessary variables and observations.
PROPOSED EXPERIMENT: Run code, including a PROC SORT, using test data containing various numbers of fields and increasing large number of observations.
THE CODE:
THE RESULTS:

CODING TIP #2:

When only one condition can be true, write a series of IF-THEN/ELSE statements.
PROPOSED EXPERIMENT: Run code, using test data containing various number of fields and increasing large number of observations, comparing IF/THEN/ELSE to WHERE to CASE to WHEN..
THE CODE:
THE RESULTS:

CODING TIP #3:

Assign a value to a constant only once
PROPOSED EXPERIMENT: Run code, using test data containing various numbers of fields and increasing large number of observations, comparing an initial RETAIN to an equation in an IF _N_=1 clause to an assignment statement performed in each iteration.
THE CODE:
THE RESULTS:

CODING TIP #4:

Consider alternatives to IF/THEN/ELSE when assigning variables, such as PROC FORMAT and MERGE.
PROPOSED EXPERIMENT: Run code, using test data containing various numbers of fields and increasing large number of observations, showing possible alternatives to an IF/THEN/ELSE construct.
THE CODE:
THE RESULTS:

CODING TIP #5:

Use the IN operator rather than logical OR operators.
PROPOSED EXPERIMENT: Run code, using test data containing various number of fields and increasing large number of observations, comparing a series of IF / THEN / ELSE statements using IN( ) vs. X=<value1> or X=<value2> ….
THE CODE:
THE RESULTS:

CODING TIP #6:

Use a series of If-THEN clauses rather than compound expressions with AND
PROPOSED EXPERIMENT: Run code, using test data containing various number of fields and increasing large number of observations, comparing an initial RETAIN to an equation in an IF _N_=1 clause to an assignment statement performed in each iteration.
THE CODE:
THE RESULTS:

CODING TIP #7:

If several different subsets are needed, avoid rereading the data for each subset**.**
PROPOSED EXPERIMENT: Run code, using test data containing various number of fields and increasing large number of observations, comparing the definition of multiple outputs on a DATA statement vs. the execution of each in a separate cloned (or Macro-ed) DATA step.
THE CODE:
THE RESULTS:

CODING TIP #8:
Read only the fields you need.
PROPOSED EXPERIMENT: Run code, using test data containing various number of fields and increasing large number of observations, demonstrating how ignoring unneeded fields (in both SET statements and INPUT statements) differ in execution.
THE CODE:
THE RESULTS:

CODING TIP #9:
Store only the variables you need.
PROPOSED EXPERIMENT: Run code, using test data containing various number of fields and increasing large number of observations, demonstrating how ignoring unneeded fields (in both DATA statements and PUT statements) differ in execution.
THE CODE:
THE RESULTS:

CODING TIP #10:
Use a null data set for writing external files.
PROPOSED EXPERIMENT:
Run code, using test data containing various number of fields and increasing large number of observations, comparing the differences between having a defined dataset in the DATA statement vs. DATA _NULL_..
THE CODE:
THE RESULTS:

CODING TIP #11:
Use MODIFY instead of SET to modify existing variables or observations in a SAS data set.
PROPOSED EXPERIMENT: Run code, using test data containing various number of fields and increasing large number of observations, demonstrating the difference between using SET vs. MODIFY. Discuss the potential risks of using MODIFY. The UPDATE statement should also be included in comparison tests.
THE CODE:
THE RESULTS:

CODING TIP #12:
Use the LENGTH statement to reduce storage space for variables in SAS data sets
PROPOSED EXPERIMENT: Run code, using test data containing various number of fields and increasing large number of observations, demonstrating how changing the length of both character and numeric fields can affect external storage AND execution time.
THE CODE:
THE RESULTS:

CODING TIP #13:
Be judicious in determining when to use MERGE vs. an SQL JOIN.
PROPOSED EXPERIMENT: Run code, using test data containing various number of fields and increasing large number of observations, demonstrating the results of MERGE vs. JOINs (Right, Left, Inner, Outer) on both sorted and unsorted input.
THE CODE:
THE RESULTS:

OTHER THOUGHTS

In several of our test cases, the anticipated results match the anticipated ones so closely that one may wonder why it was necessary to state and run the experiment. There were several reasons:

- Everyone can cite examples of popular wisdom that turned out not to be so wise – and in fact, were incorrect or even contradictory.
- The magnitude of difference was also important. One may be able to state with certainty that "Approach X is obviously better than Approach Y", but the degree to which that statement proved true may be surprising.
- In some cases, things that seem perfectly obvious to one person may be news to another. Further, a single statement may inspire a reader to pursue an entirely different line of thinking independent of the original experiment.

It is also important to note that this experiment does not factor in human effort. Some techniques can be difficult to code and test, and remain difficult to maintain after they have been rolled into a production environment. A slight improvement in thru-put that is achieved via logic that takes significant time to code and maintain may prove not to be worth the additional effort.

CONCLUSION

Our intent is NOT to make definitive statements claiming that "you should ALWAYS do something a certain way". We can state that in our environments, certain behaviors and trends were observed when similarly sized data were passed through a routine. We certainly hope that the results of our experiments prove consistent in your environment.

We DO hope that our concept of trying things out using different techniques and different volumes of data helps to illustrate how decisions can be made for coding decisions. It should be noted that the amount and volume of data to be processed, and the number of times that the process in question is to be executed, are major considerations into how much effort is put into this kind of exploration. A routine that is designed to run daily against millions of records with constraints on delivery time would be more effective with this kind of testing than would be a one-shot ad hoc report; it might take more time to do the experimentation than it would to actually code and deliver using a less-efficient mechanism!

REFERENCES / RECOMMENDED READING

Howard, Neil (2004). "How SAS® Thinks or Why the DATA Step Does What It Does". *Proceedings of the Twenty-Ninth Annual SAS Users Group International Conference.* Cary, NC: SAS Institute, Inc. http://www2.sas.com/proceedings/sugi29/252-29.pdf

Mendez, Lisa and Alonzo, Lizette (2011). "SAS® WOW! How to Streamline Your SAS Programs by Shedding Lines and Adding Substance". *Proceedings of the SAS Global Forum 2011 Conference.* Cary, NC: SAS Institute, Inc. http://support.sas.com/resources/papers/proceedings11/060-2011.pdf

SAS Institute, Inc. (2015), *SAS 9.4 Language Reference: Concepts, Fifth Edition.* Cary, NC: SAS Institute, Inc. https://support.sas.com/documentation/cdl/en/lrcon/68089/HTML/default/viewer.htm#titlepage.htm

SAS Institute, Inc. (2015), *SAS 9.4 Statements: Reference, Fourth Edition.* Cary, NC: SAS Institute, Inc. https://support.sas.com/documentation/cdl/en/lestmtsref/68024/HTML/default/viewer.htm#titlepage.htm

ACKNOWLEDGMENTS

Special thanks go out to the individuals who believed in the concept of this paper and asked that it be fleshed out and presented. In addition, our gratitude goes out to our respective employers and co-workers, as well as our families, for support while we designed and executed the experiments which are documented in this presentation.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at:

Andrew Kuligowski
HSN
St. Petersburg, FL
E-mail: KuligowskiConference@gmail.com

Swati Agarwal
Eden Prairie, MN
E-mail: epswati@yahoo.com