

How to Make Any SAS Program a Macro

Clarence Wm. Jackson

This paper details by example how to take a SAS program and make it a SAS Macro. Included will be keys to what to change, where to use macro variables, and how to make it all work together. Programs that change frequently are perfect for making it a macro. This paper is not a macro language tutorial, and assumes the reader has an understanding of the macro language, syntax, and structure.

The SAS Macro Language is included with the Base SAS System, and is really (to me) the most powerful feature of the SAS System. With it, you can do all kinds of things that aren't possible with straight SAS code. Writing macros isn't that hard, if you understand the rules of the language. As with any part of the SAS System, you can tailor it to your needs, and have it do a lot of work for you.

The example program below is for a fictional company, OK Sales, consisting of 6 sales regions called A, B, C, D, E, and F. A SAS program written in region A to list the top 25 sales people, that consists of the following code:

```
FILENAME CURSALES "OKSALES.SALES.CURRENT.TOTALS";
DATA SALES;
  INFILE CURSALES;
  INPUT @1 REGION $1. @;
  IF REGION="A" THEN DO;
    INPUT @2 SALESPER $30. @80 SOLDOLLR 9.2 @110 NUMUNITS 9.;
  END;
  ELSE DO;
    INPUT; DELETE; RETURN;
  END;
RUN;
PROC SORT; BY SOLDOLLR; RUN;
TITLE "OK SALES";
TITLE2 "REGION A'S TOP 25 SALES BY DOLLARS";
DATA TOPSALES;
  SET SALES;
  BY SOLDOLLR;
  X+1;
  IF X>25 THEN DO;
    FILE PRINT;
    ///@ 25 SALESPER $30. @ 65 SOLDOLLR COMMA21.2 @95 NUMUNITS COMMA19.;
    STOP;
  END;
RUN;
ENDSAS;
```

Region A's management used the output of the report to boost production in sales, and it worked so well, that Region B wanted the report for their people. To make things better (or worse), there were rumors about how Region A was exceeding because of that SAS report, Headquarters was reviewing the situation, and might want the SAS report available for all the regions. Changes are on the way, and the SAS programmer who wrote the code is

wondering what to do next, and how to avoid a major investment of time and resources to take his SAS program to the next level.

The SAS programmer had to make a few decisions about how to change the program. Should he (a) make a separate version for each region, (b) modify the code to include "IF" clauses for each region to get the reports, or (c) modify the program to do all the regions? How do you make the change to the program so that all Regions can get their reports, allow for future expansion, and not copy and paste the program or make multiple copies?

One option is to convert the program into a macro. This is done by reviewing the program for code that can become Macro Variables, and making the changes to call the program. To make it dynamic, an extra element is added to review data passed to the macro. An example of a set of changes for the code follows:

```
%MACRO TOP25 (FOREGION); * Start the macro definition using positional parm to pass the region id;
DATA _NULL_; * Extra step added to make the macro dynamic;
IF _N_ EQ 1 THEN DO; * Setup macro variables for reports by region;
  CNT=LENGTH(&REGION); * Get the length of the macro parameter string;
  CALL SYMPUT ('NUMIN',CNT); * Put the number of regions into a macro variable;
END;
REG=SUBSTR(&REGION,CNT,1);
CALL SYMPUT ('REGION'!!TRIM(LEFT(PUT(CNT,2.))),REG); * Put each region into a variable
CNT-1; * Decrement the total count of regions;
RUN;
FILENAME CURSALES "OKSALES.SALES.CURRENT.TOTALS";
%IF &NUMIN=0 THEN %GOTO EXITOUT; * Check
%DO NUMX=1 %TO &NUMIN; * Execute for each region in the parm list;
DATA SALES;
  INFILE CURSALES;
  INPUT @1 REGION $1. @;
  IF REGION=&&REGION&NUMX THEN DO;
    INPUT @2 SALESPER $30. @80 SOLDOLLR 9.2 @110 NUMUNITS 9.;
  END;
  ELSE DO;
    INPUT; DELETE; RETURN;
  END;
RUN;
PROC SORT; BY SOLDOLLR; RUN;
TITLE "OK SALES";
TITLE2 "REGION &&REGION&NUMX'S TOP 25 SALES BY DOLLARS";
DATA TOPSALES;
  SET SALES;
  BY SOLDOLLR;
  X+1;
  IF X>25 THEN DO;
    FILE PRINT;
    ///@ 25 SALESPER $30. @ 85 SOLDOLLR COMMA21.2 @95 NUMUNITS COMMA19.;
  STOP;
RUN;
%END;
%EXITOUT:
%MEND TOP25; * End of the Macro definition;
%TOP25(AB); * Call the TOP25 macro to oreate the reports for regions in the list;
ENDSAS;
```

The changes made to the code looks like a lot, but the result of the changes is what matters. Now, the programmer will never need to modify the program to include additional regions, even those not yet in existence. Only the "Call" to the macro will need to be modified, and if the macro was put to an AutoCall library, then other SAS programmers in the other regions (like region E) could also use it by using the following in their SAS program:

```
%TOP25 (E);
```

So what was accomplished?

- We avoided having to make a separate version (or copy) of the code for the new region. The one program processes all regions
- We made the program dynamic. We don't have to go into the guts of the program to modify it.
- We may never need to add code for the new regions. We can use the program as is in the event of expansion into regions H-Z without modifications to the programs. As long as the data is there, the program should run.
- We have made the macro self modifiable. We only supply the region's ID, and the program selects the correct data for it, and changes the titles to include the region ID.

I wouldn't recommend making every SAS program a macro, but there are some that would work better as macros. For each program, you'll need to ask some basic questions, such as

- What's going to change in the program that will require modification? For example, reports based on date periods may require you to supply the start and end dates. Excellent candidate for a macro.
- What modifications must be made each time the program is executed? Do you need to change the title, PROCedure type, or other changes depending on external information?
- What modifications to the program, logic, or other controls in the program can be made via the macro facility? SAS has very good tools to determine what and where things are in your program, data, and operating system environment that can be accessed via the macro language facility thru the automatic macro variables.
- What modifications must be made because of data sources, data elements, or data values? You can include conditional %IF %THEN %ELSE to have totally self modifying SAS programs, skipping major parts of SAS programs based on the data information.
- How much of a pain is this going to be? If the program you are thinking about changing is used once a year and requires minor or no modifications, or is modified infrequently, it might be a waste of time. However, if it is a frequently executed "Production" type program that has to be changed before each execution, you might want to modify it. You should look at all the changes you're making to the program, and make those macro aware. Why waste your time changing a program each time you run it when you can change it once, and run it many times?

Any program can be modified to be a macro, the decision should be based on how much value can be gained by making it a macro. If you get multiple requests from multiple people regarding the same program "only with my data", the program could be a candidate for making a macro.

Other Sources of Information can be found in the *SAS Language Reference*, *SAS Macro Guide*, and other books, such as Art Carpenter's *Complete Guide to the SAS Macro Language*, on the subject of macros.

Take care, and understand the macro language before trying things out. The things you can do with macros are powerful, and can also do damage if you're not careful. But with understanding of your problem and your program, you can make any SAS program a SAS macro.

About the Author

Clarence Wm. Jackson currently works for the City of Dallas ECI as the Production Control Supervisor and Quality Assurance Analyst. You can reach him via the Internet at CJac@compuserve.com, cljacks@ci.dallas.tx.us, or via his homepage at <http://ourworld.compuserve.com/homepages/CJac>. Snail mail him using the City of Dallas ECI (Operations and Plans), 1500 Marilla St 4DS, Dallas, TX 75201.