Tips for Using a Data Warehouse
Curtis Alan Smith

In our last episode we explored how to create a WHERE statement dynamically based on the values in a variable in a SAS data set that we specify. This time we will explore how to conditionally execute code based on the number of observations in a SAS data set.

I frequently have an application where I will want to perform an action, like exporting results in a SAS data set to Excel, but only if there are results. Sometimes the results of my application will result in an empty SAS data set. When that happens, I do not want my application to go through the motions of trying to export an empty SAS data set to Excel. So, I need to do two things: first, capture the number of observations in my target SAS data set, and second, execute an action conditionally based on the number of observations.

Using my sands.mysasdata SAS data set, I want to capture the number of observations into a macro variable that I will call *numobs*. To do so, I will use the following DATA _NULL_ step:

```
data _null_;
  set sands.mysasdata nobs=n;
  call symputx('numobs',n);
  stop;
run;
%put NOTE: Number of rows in sands.mysasdata = &numobs;
```

Here, *numobs* is a macro variable I named that will hold the number of observations in the SAS data set. But, how do I get the number of observations into *numobs*? We will use the NOBS= option to first store the number of observations in our input SS data set into a new data set variable, which we will call simply *n*. Let's get some background on the NOBS= option from the SAS Institute (Sas® 9.4 data step statements, 2017):

> *NOBS=variable*
> *creates and names a temporary variable whose value is usually the total number of observations in the input data set or data sets. If more than one data set is listed in the SET statement, NOBS= the total number of observations in the data sets that are listed. The number of observations includes those observations that are marked for deletion but are not yet deleted.*

Now I have a variable containing the number of variables in my sands.mysasdata SAS data set. But, the variable only exists in the NULL data set, so I need to get the value of that variable into something that I can use later. For my purposes, a macro variable will do the trick. So, how to get the value of the new variable *n* into a macro variable? The number of observations in variable *n* is a numeric value and the macro variable that I want to create will only hold a character value. So, I also need to convert the numeric

value to character before stuffing it into a macro variable. CALL SYMPUTX Routine to the rescue! The CALL SYMPUTX Routine automatically converts the numeric variables value to a character before assigning it to a macro variable. This eliminates the need to first convert the numeric value using a PUT statement. Let's get some background on the CALL SYMPUTX Routine from the SAS Institute (Sas® 9.4 macro language: reference, 2017):

> *CALL SYMPUTX*
> *Assigns a value to a macro variable, and removes both leading and trailing blanks.*
>
> *CALL SYMPUTX left-justifies both arguments and trims trailing blanks.*

A nice thing about NOBS= is that it gets the number of observations from the metadata so the DATA step does not need to process every row. So, once we have the number of observations we need from the metadata, we want the DATA step to stop. To do so, we will use the STOP statement. Let's get some background on the STOP statement from the SAS Institute (Sas® 9.4 data step statements, 2017b):

> *STOP Statement*
> *The STOP statement causes SAS to stop processing the current DATA step immediately and resume processing statements after the end of the current DATA step.*

When I execute this code, the log will look like the following:

```
84   data _null_;
85     set sands.mysasdata nobs=n;
86     call symputx('numobs',n);
87     stop;
88   run;

NOTE: There were 1 observations read from the data set
SANDS.SANDS_2016_DATA.
NOTE: DATA statement used (Total process time):
      real time           0.01 seconds
      cpu time            0.01 seconds

89   %put Note: Number of rows in sands.mysasdata = &numobs;
Note: Number of rows in sands.mysasdata = 3327854
```

Now that I have a macro variable containing the number of rows in the SAS data set, how can I use it? For the answer, tune in next time. Just kidding. I want to conditionally execute a macro in which I have instructions to export the SAS data set to Excel. But, the statements that I want to use cannot be executed in open code, so I will place the conditional processing in its own macro.

```
%macro export_data;
%if &numobs. > 0 %then %do;
     %exportxlsx;
%end;
%else %do;
     %put NOTE: No rows to export.;
%end;
%mend export_data;
%export_data;
```

After I compile the %macro export_data and execute it, the %IF statement will compare the value stored in &numobs. to 0 and if true, then execute the macro %exportxlsx. It does not matter that the value in the macro variable is stored as a character which is compared to the numeral 0 because the macro variable will resolve when the macro is executed and the marco variable token will be replaced with the value stored in it. Which in this case is 3327854, which of course is greater than 0. So, the SAS data set will be exported to Excel. But, whenever the application results in sands.mysasdata having no observations, then the %IF statement will be false and the %ELSE statement will process, which will place a message in the log.

Next time we will explore...well, I'm not sure what we will explore, but it will be interesting.

Thanks for reading.

Curtis A. Smith
DoD Program Manager
ca.smith86@att.net

(2017) SAS® 9.4 DATA Step Statements: Reference; SET Statement. Cary, NC: SAS Institute, Inc. Retrieved from http://documentation.sas.com/?docsetId=lestmtsref&docsetTarget=p00hxg3x8lwivcn1f0e9axziw57y.htm&docsetVersion=9.4&locale=en

(2017) SAS® 9.4 Macro Language: Reference, Fifth Edition; CALL SYMPUTX Routine. Cary, NC: SAS Institute, Inc. Retrieved from http://documentation.sas.com/?docsetId=mcrolref&docsetVersion=9.4&docsetTarget=p1fa0ay5pzr9yun1mvqxv8ipzd4d.htm&locale=en

(2017) SAS® 9.4 DATA Step Statements: Reference; STOP Statement. Cary, NC: SAS Institute, Inc. Retrieved from http://documentation.sas.com/?docsetId=lestmtsref&docsetTarget=p02jy612cym9oxn10i19cbewlrdf.htm&docsetVersion=9.4&locale=en