Paper 067-2011

Automatically Identify Formatting Problems with Numeric Variables

Wei Xu, Boston Scientific, Marlborough, Massachusetts

ABSTRACT

Have you ever thought about how to identify formatting problems with numeric variables? When you analyze all formatted numeric variables in a data set and identify any variable whose entire range of values cannot be displayed with the assigned format. These variables cause SAS to produce the following message: At least one W.D format was too small for the number to be printed.

This paper will describe a simple straightforward way to write a message to the SAS log for all problematic variables. The macro GETLST uses PROC CONTENTS to generate a data set containing the format width and decimal values for each formatted numeric variable. A subsequent DATA step stores this information in macro variables. PROC SQL is then used to determine the magnitude of the minimum and maximum values of each variable. This information, as well as Boolean indicators for positive and negative values is stored in macro variables. The macro then checks whether each variable's entire range of values can be displayed with the assigned format. If the format's width is insufficient, a message is written to the SAS log. A message “No formatted variables in the dataset” is also generated in the specified data set if there are no qualifying variables.

THE SETUP

You will need a few things to get this set up.

1. Create the project location of the %GETLST macro definition file on your system.
2. In your SAS® program or in the SAS editor window, specify the statement to define the %GETLST macro and make it available for use.
3. Following the statement, you need to call the %GETLST macro.

THE MACRO %GETLST

Specify this statement to define the %GETLST macro and make it available for use:

```
/*************************************************************************************************************************************/

 %include "<location of your file containing the GETLST macro>";

 %getlst (library, member)

/*************************************************************************************************************************************/
```

THE EXAMPLE:

The macro GETLST uses PROC CONTENTS to generate a data set containing the format width and decimal values for each formatted numeric variable. A subsequent DATA step stores this information in macro variables. PROC SQL is then used to determine the magnitude of the minimum and maximum values of each variable. This information, as well as Boolean indicators for positive and negative values are stored in macro variables. The macro then checks whether each variable's entire range of values can be displayed with the assigned format. If the format's width is insufficient, a message is written to the SAS log. A message “No formatted variables in the dataset” is also generated in the specified data set.

```
/*************************************************************************************************************************************/

/* Create a data set with a formatted numeric variable. */

data abc;
   x=-1234;
   format x 4.1;
   put x=;
run;
```

```
%macro getlst(library,member);

 proc contents noprint data=&library..&member.
      out=work.cntsout(keep=name type formatl formatd
      where=(type=1 and formatl > 0));
 run;

/* This DATA step reads the data set generated by CONTENTS and creates three macro
variables for each formatted numeric variable in LIBRARY.MEMBER.  It also creates the
macro variable VARCNT with the total number of variables. */

 %let varcnt=0;
  data _null_;
     set work.cntsout end=done;
     call symput('varname'||put(_n_,5. -1),name);
     call symput('width'||put(_n_,5. -1),put(formatl,2. -1));
     call symput('decimal'||put(_n_,5. -1),put(formatd,2. -1));
     if done then call symput('varcnt',put(_n_,5. -1));
  run;

/* If there are variables to process, continue. */

  %if (&varcnt ne 0) %then %do;

/* Use PROC SQL to find the magnitude of the minimum and maximum values of each
variable, create Boolean indicators for the presence of positive and negative values,
and store these in macro variables. */

      proc sql noprint;
         %do i = 1 %to &varcnt;
            select scan(put(max(&&varname&i),e.),2,'+-'),
                   scan(put(min(&&varname&i),e.),2,'+-'),
                   put((max(&&varname&i) ge 0),best.-1),
                   put((min(&&varname&i) lt 0),best.-1)
            into :pos_expon&i, :neg_expon&i, :pos_values&i, :neg_values&i
            from &library..&member;
         %end;
      quit;

/* Determine the smallest format width required to display the variable's entire range
of values and compare that to the width of the assigned format. If the width is
insufficient, write a message to the log. */

      %do i = 1 %to &varcnt;
       %let magic_number=%sysfunc(max(%eval(&&neg_expon&i+&&neg_values&i+1),
                                  %eval(&&pos_expon&i+&&pos_values&i)));
       %if ((&&decimal&i eq 0 or &&decimal&i eq ) and
           (&&width&i lt &magic_number)) or
           (%eval(&&width&i - &&decimal&i -1) lt &magic_number)
```

```
        %then %put problems with %trim(&&varname&i) and
              format %trim(&&width&i).&&decimal&i;
      %end;
    %end;

/* Write a message if there are no qualifying variables in the data set. */

   %else %put No formatted variables in the data set &library..&member;
%mend;

%getlst(work,abc)

/**********************************************************************************************************************************/
```

THE RESULTS

The program writes a message to the SAS log for all problematic variables:

problems with X and format 4.1

CONCLUSION

This is a simple way to identify formatting problems with numeric variables.

The advantages are:
(1) The new macro can be created to automatically identify formatting problems with numeric variables.
(2) This macro %GETLST should be located in a central place such as project macro area and each project can call the macro from there.

ACKNOWLEDGEMENTS

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.
Other brand and product names are trademarks of their respective companies.

REFERENCE

[1] SAS Institute, Inc., Cary, NC, SAS® Guide to Macro Processing: Version 6, Second Edition.
[2] SAS Institute, Inc., Cary, NC, SAS® Procedures Guide: Version 6, Third Edition.
[3] SAS Institute, Inc., Cary, NC, SAS® Language Reference: Version 6, First Edition.
[4] SAS Institute, Inc., Cary, NC, SAS® Language Reference: Version 6, Second Edition.

CONTACT INFORMATION

Wei Xu
Boston Scientific
100 Boston Scientific Way
Marlborough, MA 01752-1234
(508) 683-4264
wei.xu@bsci.com