Creating Macro Calls using Proc Freq

Wendi L. Wright, Educational Testing Service, Princeton, NJ

ABSTRACT

Imagine you were asked to get a series of statistics/tables for each country in the world. You have the data, but when you run a distribution of country you find you have 680 countries. Assume at this point that using different SAS procedures with the 'by' statement will not give you what the client wants.

Obviously the easiest method is to create a macro to get the statistics and call the macro once for each country. However, writing the macro calls for all 680 countries would be very tedious. And the next time you are asked to run this, the countries may be different and you would need to retype many of the macro calls.

There is a way to use macro language and a proc freq to generate the macro calls you need. The technique is also very useful if you need to run statistics for levels of several demographic variables (e.g. country, income level, educational level, etc.)

This paper will demonstrate the use of macro language and a proc freq to generate a macro call for each level of a demographic variable without typing each level and without foreknowledge of exactly what levels exist in the data.

INTRODUCTION

The ‘repeat’ macro (see Appendix A) generates calls to the ‘ctable’ macro (see Appendix B) that runs an analysis and produces a report for each value of a demographic variable. It includes a frequency distribution with percent below and summary statistics (see Appendix C). The called macro (‘ctable’) and the sample report are provided only as an example of how the ‘repeat’ macro can be used.

THE BASICS

I used several standard SAS techniques in this macro. Among them are:

1. PROC FREQ will create a SAS data set with the results of a frequency distribution using the OUT= option on the TABLES statement.
2. Macro variable substitution using a variable name prefixed with ‘&’.
3. The SAS interpreter interprets a ‘&&’ as a single ‘&’ when it first reads the program. This allows the program to create the macro variables and values before the interpreter attempts to resolve them.

HOW THE REPEAT MACRO WORKS

To simplify my explanation I created a sample dataset with two variables (score and educational level). The by variable (byvar) that I used is edlevel. Edlevel has six possible values. The macro call will look like this: %repeat(edlevel)

When the macro executes, the first step is a PROC FREQ. This creates an output data set called ‘list’ that has the variables edlevel, count and percent. Note these values are unformatted and that the value of ‘Jun’ stands for ‘Juniors’. Here are a few records from a print of the data set ‘list’:

<table>
<thead>
<tr>
<th>OBS</th>
<th>Edlevel</th>
<th>Count</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Soph</td>
<td>202</td>
<td>15</td>
</tr>
<tr>
<td>5</td>
<td>Jun</td>
<td>281</td>
<td>21</td>
</tr>
<tr>
<td>6</td>
<td>Sen</td>
<td>306</td>
<td>23</td>
</tr>
</tbody>
</table>

The data step reads in this output data set and does a couple of things. First it creates a macro variable called ‘totain’ that holds the total number of observations in the dataset ‘list’. In this example, the value of totain is ‘6’.

The second part of the data step creates a macro variable for each observation of the data set. Each macro variable holds the value of the byvar (edlevel) variable for that observation. For example, for the fifth observation of the ‘list’ data set, the macro variable created would be x5 and would hold a value of ‘Jun’.

The last part of the macro: (1) Verifies there were records in the output data set ‘list’ (2) runs a loop starting with observation 1 to the maximum number of observations in the data set; (3) creates a macro call for each iteration of the loop that calls another macro ‘ctable’. For example, the macro call generated for the level of Juniors would look like:

%ctable (edlevel,Jun)

The output from the ‘Juniors’ run of the ‘ctable’ macro is in Appendix C.

CONCLUSION

The ‘repeat’ macro is rather short, but very powerful. It can be used to quickly and easily produce multiple reports based on any demographic criteria without the need for a large amount of typing.

AUTHOR CONTACT

Your comments and questions are valued and welcome. Contact the author at:

Wendi L. Wright
Educational Testing Service, 02-L
Princeton, NJ 08541
Phone: (609) 683-2292
Fax: (609) 683-2130
E-mail: wwright@ets.org
APPENDIX A
(the 'Repeat' Macro)

%macro repeat (byvar);
  /*
   Wendi L. Wright
   Educational Testing Service
   June 2001
   */

  *CALCULATE A LIST OF ALL LEVELS OF THE DEMOGRAPHIC VARIABLE;
  proc freq data=one;
    tables &byvar / out=list;
  run;

  *CALCULATE TWO TYPES OF MACRO VARIABLES:
    1. TOTALN (one macro variable)= has the number of observations in the
       'list' dataset ( or the number of different values found in the
       demographic variable.
    2. X# - has one macro variable for each observation in the dataset 'list'.
       Will hold the value of the demographic variable that occurs in that
       observation.
  ;
  data list;
    set list nobs=numobs;
    if _n_ eq 1 then call symput ('totaln',left(numobs));
    call symput ('x'||left(_n_),&byvar);
  run;

  *Using the TOTALN and the 'X#' macro variables created above, call the macro
    ctable once for each level of the by-variable.;
  %if &totaln ne 0 %then %do;
    %do j=1 %to &totaln;
      %ctable (&byvar,&&x&j)
    %end;
  %end;
  %mend;
APPENDIX B
(the 'CTable' Macro)

%MACRO ctable (var,value);
   /*
      Wendi L. Wright
      Educational Testing Service
      June 2001
   */
   *
   * CREATE A MACRO VARIABLE WITH A SAMPLE DESCRIPTION - TO BE
   INCLUDED IN THE TITLE.
   NOTE: You will need a previously defined format for the
   'var' variable. For example, if your variable is named
   'edlevel', then you will need a format name $edlevel.;

   data temp;
      set one;
      if &var eq "&value" then do;
         call symput('samp',trim(put(&var,$&var..)));
         output;
      end;
   run;

   *CALCULATE THE FREQ DISTRIBUTION, AND THE PERCENT BELOW;
   proc freq data=temp;
      tables score / out=a noprint;
   run;
   PROC SORT DATA = a;
      BY descending score;
   *CALCULATE THE N, MEAN, AND THE STANDARD DEVIATION;
   proc means data=temp vardef=n;
      var score;
      output out=b mean=mean std=std n=n;
   * PRINT OUT THE FREQUENCY DISTRIBUTION AND THE SUMMARY STATS ;
   data c;
      set a;
      if _n_ eq 1 then set b;
   run;

   data c;
      set c end=eof;
      file out1 ps=60 n=ps header=h mod notitle;
      if n ge 10 then do;
         put @31 score 3.
         @38 count 6.
         @50 pbelow 4.1
      ;
if eof then do;
   put /// @25 '*************************************' / 
       @35 'N'
       @41 'Mean'
       @51 'STD'
       / @30 n 6.
       @42 mean 3.
       @51 std 3.
       / @25 '*************************************' / 
;
end;
end;
return;

*TITLE THAT GOES ON EVERY PAGE ;

h:
   put @30 'Summary Statistics' / 
       @30 'Data for Fiscal Year 2000-01' / 
       @30 "%samp Examinees" //
       @30 'Score'
       @40 'Count'
       @49 '% Below' / ;
return;
run;

%MEND;
Summary Statistics
Data for Fiscal Year 2000-01
Edlevel = Junior Examinees

<table>
<thead>
<tr>
<th>Score</th>
<th>Count</th>
<th>%Below</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>15</td>
<td>95</td>
</tr>
<tr>
<td>198</td>
<td>2</td>
<td>94</td>
</tr>
<tr>
<td>192</td>
<td>5</td>
<td>92</td>
</tr>
<tr>
<td>180</td>
<td>25</td>
<td>83</td>
</tr>
<tr>
<td>172</td>
<td>51</td>
<td>65</td>
</tr>
<tr>
<td>165</td>
<td>78</td>
<td>37</td>
</tr>
<tr>
<td>158</td>
<td>44</td>
<td>22</td>
</tr>
<tr>
<td>145</td>
<td>32</td>
<td>10</td>
</tr>
<tr>
<td>137</td>
<td>14</td>
<td>5</td>
</tr>
<tr>
<td>122</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>114</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>101</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

******************************************
N  Mean  STD
281 163  91
******************************************