

Automatic Creating SAS® Macros to Manage Unknown Values in Year/Month Variables for Year 2000 Issue

Jianmin Li, TRW Inc., Atlanta, GA
Shelley X. Ren, TRW Inc., Atlanta, GA
Pei-Chun T. Wan, TRW Inc., Atlanta, GA

ABSTRACT

This paper illustrates how one can use PUT statement, IF-THEN statement, and CALL SYMPUT routine to automatically create SAS Macro variables and Macro statements, and then use ARRAY processing and DO loops to summarize and convert unknown values in year/month variables to prepare for Year 2000 issue.

In the National HIV/AIDS Surveillance database at Centers for Diseases Control and Prevention (CDC), "99" is used to code unknown value for year/month variable. To prepare for year 1999, character "99"s for unknown years are needed to be converted to character "..".

By using the technique described herein, one can use CONTENTS procedure to output month/year variable names to a SAS data set, use PUT statements to automatically create macro variables, and use FILE and %INCLUDE statements to place and locate files with list of macro variables and macro statements. In addition, CALL SYMPUT routine is used to count the number of year/month variables to automatically define ARRAY dimension for summarizing and converting the unknown values in year/month variables using DO Loops.

INTRODUCTION

There are about 600 variables in the National HIV/AIDS Surveillance SAS database at CDC.

More than one third of those variables are dates in the form of year and month (YRMO) associated with diagnosis, reporting, lab testing results, etc. Currently, "99" is used to code unknown year or month, and "9999" is used to code unknown YRMO.

The problem arises when the data of 1999 will be incorporated into the surveillance system. For example, will the first two digits in a value of "9999" for a YRMO variable represent year 1999 or a missing year? The same question would also present for years after 2000. Thus, to prepare for year 1999 and the coming century, it is necessary to convert the missing values from "99" to ".." in the surveillance system. To make this conversion process successful, the first step is to evaluate the magnitude of the "99" in each YRMO variable so appropriate procedures can be applied to all possible cases.

The program presented in this article is for this evaluation purpose. The use of the program can be extended to the examination of the quality of conversion. Furthermore, this program can be applied to other situations when similar problems need to be resolved.

The program is completed by using the CONTENTS procedure to output YRMO variables to a SAS data set, the PUT statements to automatically create macro variables, and the FILE and %INCLUDE statements to place and locate files with list of macro variables and macro statements. In addition, CALL SYMPUT routine is used to count the number of

CODERS CORNER

year/month variables to automatically define ARRAY dimension for summarizing and converting the unknown values in year/month variables using DO Loops.

METHOD

A complete program source code for this paper is presented in Appendix A. Important elements of the program are highlighted here for explanation.

Automatically Create User-defined Macro Variables

The first part of the program is to select variable names that contain character string of YRMO and then to store those variables into a SAS Macro variable **string** as follows:

```
%let string=ab_yrmo
             bactyrmo
             .
             .
             .
             _rhvyrmo
             _th_yrmo;
```

The following codes automatically create **string** by using IF-THEN/ELSE, FIRST., LAST., and PUT statement and store the Macro variable in an external file. The program also counts the number of variables stored in **strings** by using CALL SYMPUT routine, and store it in a Macro variable **numvar**.

```
data variable;
  set variable;
  num=1;

data _null_ ;
  set variable;
  by num;
```

```
file "&macrostr";
```

```
if first.num then do;
  put @1 ' %let string= ' @14 name $8.;
end;
```

```
else if not last.num then do;
  put @14 name $8.;
end;
```

```
else if last.num then do;
  put @14 name $8. @22 ' ';
  call symput('numvar', _n_);
end;
```

```
run;
```

Automatically Create SAS Statements for Calling Macros

To compute percent of missing YRMOs for each of the YRMO variable, a Macro %freq was defined. To call the %freq for each variable, a set of statements need to be created as follows:

```
%freq (ab_yrmo );
%freq (arv_yrmo);
.
.
.
%freq (_rhvyrmo);
%freq (_th_yrmo);
```

The following codes automatically create the set of statements by using PUT statement, and store the set of statements in an external file.

```
data _null_ ;
  set variable;
  file "&macrogrp";
  put @1 ' %freq ( ' @20 name $8. @28 ' );' ;

run;
```

Reference Macro Variables and Examination of "9999"s

The following codes reference the macro variable **string** and **numvar**:

```
%include "&macrostr";

data yrmo;
  set &dataset (keep= &string);

  array miss (&numvar) &string;
```

Using DO loop to exam "9999" in the YRMO variables:

```
do i=1 to &numvar;

  if substr(miss(i),1,2)='99'
    and substr(miss(i),3,2) not in ('99',' ')
    then miss(i)='1'; else

  if substr(miss(i),3,2)='99'
    and substr(miss(i),1,2) not in ('99',' ')
    then miss(i)='2'; else

  if miss(i)='9999'
    then miss(i)='3'; else

  if miss(i)=' '
    then miss(i)='5'; else
    miss(i)='4';

end;
```

Define and Call Macro %FREQ

```
%macro freq (var);

proc freq data=yrmo;
  tables &var / missing noprint out=out;

data out (keep=variable count percent);
```

```
set out;
length variable $8.;
percent=percent/100;
if &var ne '1' then delete;
variable="&var";
```

```
data a;
  set a out;
```

```
%mend;
```

```
data a;
```

```
%include "&macrogrp";
```

CONCLUSION

This technique can be applied to programs which require user to select a subset of variables with similar elements in their names, and manage the variables with SAS Macro, Array, and Do loops. The major advantage of the technique is to keep the program flexible in working with any giving number of variables without change the source codes.

REFERENCES

SAS Institute Inc., SAS Guide to Macro Processing, Cary, NC: SAS Institute, Inc., 1990

SAS Institute Inc., SAS Macro Language Course Notes, Cary, NC: SAS Institute, Inc., 1993

SAS is registered trademark of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

CODERS CORNER

Contact lead author at:

16 Executive Park Drive, MS-E47
NCHSTP/DHAP/SURV BRANCH
Centers for Diseases Control and Prevention
Atlanta, GA 30329
(404) 639-2050
Email: JBL8@cdc.gov

APPENDIX A.

/* EXAMPLE OF SAS PROGRAM */

Created by: Jianmim Li

This program is written for:

Statistics and Data Management Branch
Division of HIV/AIDS Prevention
National Center for HIV, STD, and TB
Prevention

The purpose of the program is to assess the
percentage of the AIDS cases with missing
year and non-missing month for all year and
month (YRMO) variables on National AIDS
Surveillance database

**** GLOBAL MACRO VARIABLES ****;

* INPUT SAS DATA SET;

%let dataset=hars.as_feb98;

* FILE NAME FOR &STRING;

%let macrostr=l:\jbl8\saspgm\xiaoyan3.str;

* FILE NAME FOR &FREQ;

%let macrogrp=l:\jbl8\saspgm\xiaoyan3.mac;

* VARIABLE SELECTION CRITERIA;

%let select=YRMO;

options obs=max;

title1 'CDC AIDS SURVEILLANCE DATA
AS OF 12/31/97';
title3 'YRMO VARIABLES WITH 99 IN YR
AND NON-MISSING VALUE IN MO';
footnote 'Program: l:\jbl8\saspgm\xiaoyan4';

**** WRITE PROC CONTENTS OUTPUT
TO A SAS DATA SET ;

proc contents data=&dataset noprint
out=contents (keep=name);

**** KEEP VARIABLE NAMES WITH
SELECTED CHARACTER STRING;

data variable;
set contents (where=
(name contains "&SELECT"));

**** CREATE A MACRO VARIABLE TO
STORE ALL SELECTED VARIABLES AND
A MACRO VARIABLE TO COUNT
NUMBER OF THE VARIABLES
SELECTED;

data variable;
set variable;
num=1;

data _null_ ;
set variable;
by num;
file "¯ostr";
if first.num then do;
put @1 ' %let string= ' @14 name \$8.;
end;
else if not last.num then do;
put @14 name \$8.; end;
else if last.num then do;
put @14 name \$8. @22 ' ';

```
call symput('numvar', _n_); end;
```

```
**** CREATE %FREQ STATEMENT FOR  
EACH SELECTED YRMO VARIABLE;
```

```
data _null_ ;  
  set variable;  
  file "&macrogrp";  
  put @1 '%freq (' @20 name $8. @28 ');' ;
```

```
run;
```

```
%include "&macrostr";
```

```
**** PROCESSING ALL YRMO  
VARIABLES IN AN ARRAY;
```

```
data yrmo;  
  set &dataset (keep= &string);  
  
  array miss (&numvar) &string;  
  
  do i=1 to &numvar;  
  
    if substr(miss(i),1,2)='99'  
      and substr(miss(i),3,2) not in ('99',' ')  
      then miss(i)='1'; else  
  
    if substr(miss(i),3,2)='99'  
      and substr(miss(i),1,2) not in ('99',' ')  
      then miss(i)='2'; else  
  
    if miss(i)='9999'  
      then miss(i)='3'; else  
  
    if miss(i)=' '  
      then miss(i)='5'; else  
      miss(i)='4';  
  
  end;  
  
proc format;  
  value $miss '1'='yr=99 mo=valid'
```

```
'2'='mo=99 yr=valid'  
'3'='yrmo=9999'  
'4'='yrmo=valid'  
'5'='yrmo=blank';
```

```
**** DEFINE MACRO %FREQ FOR  
ASSESSING MAGNITUDE OF VARIABLES  
WITH MISSING IN YEAR AND NON-  
MISSING IN MONTH;
```

```
%macro freq (var);
```

```
proc freq data=yrmo;  
  tables &var / missing noprint out=out;
```

```
data out (keep=variable count percent);  
  set out;  
  length variable $8.;  
  percent=percent/100;  
  if &var ne '1' then delete;  
  variable="&var";
```

```
data a;  
  set a out;
```

```
%mend;
```

```
data a;
```

```
%include "&macrogrp";
```

```
data a;  
  set a;  
  if _n_=1 then delete;
```

```
**** PRINT OUT FINAL SUMMARY  
TABLE;
```

```
proc print data=a noobs;  
  var variable count percent;  
  sum count;  
  format percent percent10.2;  
run;
```