A Simple Macro to Minimize Dataset Size

Amos Shu,
Endo Pharmaceuticals., Malvern, PA 19355

Introduction

Resizing text columns to minimize the size of a SAS dataset is not only useful for FDA submission in the pharmaceutical industry, but also across all areas that use SAS datasets to do business. It can save disk space and reduce the number of I/O operations required to read and write the dataset. This paper discusses a simple macro to minimize the size of a SAS dataset.

Step 1. Select Only Character Variables from the Dataset

SAS has a default length of 8 bytes to store numeric variables. The issue of numeric precision affects the return values of almost all SAS math functions and many numeric return values from SAS procedures, so we normally do not modify the lengths of numeric variables for just resizing purpose.

The macro has only one parameter – inp, which the name of the resized dataset will be assigned to when the macro is invoked.

```
%MACRO  ChgLen (inp=);
        … …
%MEND ChgLen;
```

The first step in the macro is to select only character variables from the dataset using PROC CONTENTS:

```
PROC CONTENTS DATA = &inp.   NOPRINT
          OUT  = CharVar (WHERE =(type=2)
     KEEP = memname name type length label);
RUN;
```

The output looks like this:

Library Member Name	Variable Name	Variable Type	Variable Length	Variable Label
ADAE	AESTDTC	2	50	Adverse Event Start Date
ADAE	AETERM	2	200	Reported Term for the Adverse Event
ADAE	AEYN	2	1	Any Adverse Event Experienced Flag
ADAE	USUBJID	2	20	Unique Subject Identifier

Step 2. Create SAS Codes by Using Length and Max Functions

The second step is to generate SAS codes that contain the length or max function in a DATA step.

```
DATA  CharVar2;
     LENGTH  name1  name2  name3  name4   name5  $100;
     SET CharVar;
     name1 = trim(name)||'1=length('||trim(name)||')';
     name2 = 'max('||trim(name)||'1)  as '||trim(name);
     name3 = trim(name)||'x = '||trim(name);
     name4 = trim(name)||' = '||trim(name)||'x';
     name5 = trim(name)||'  $'||strip(put(LENGTH, best.));
RUN;
```

The above code generates five variables - name1, name2, name3, name4, and name5, which contain values like “AETERM1=length(AETERM)”, “max(AETERM1) as AETERM”, “AETERMx = AETERM”, “AETERM = AETERMx”, “AETERM $200”, respectively.

Step 3. Create Macro Variables That Will be Used to Generate Different SAS Statements

```
PROC SQL  NOPRINT;
    SELECT  trim(name1) INTO: newvar  SEPARATED BY ';  '
    FROM CharVar2;
    SELECT trim(name2) INTO: maxvar  SEPARATED BY ',  '
    FROM CharVar2;

    SELECT trim(name3) INTO: tname  SEPARATED BY ';  '
    FROM CharVar2;

    SELECT trim(name4) INTO: tnamex  SEPARATED BY ';  '
    FROM CharVar2;

    SELECT trim(name) INTO: CVar  SEPARATED BY ' '
    FROM CharVar2;

    SELECT trim(name)||'x' INTO: CVarx  SEPARATED BY ' '
    FROM CharVar2;

    SELECT trim(name5)  INTO: CVarLen  SEPARATED BY ' '
    FROM CharVar2;
QUIT;
```

Shu -

These macro variables will resolve to the following values when they are called:

Macro Variable	Resolutions
&newvar	AESTDTC1=length(AESTDTC); AETERM1=length(AETERM); AEYN1=length(AEYN); SAFFL1=length(SAFFL); …
&maxvar	max(AESTDTC1) as AESTDTC, max(AETERM1) as AETERM, max(AEYN1) as AEYN, max(SAFFL1) as SAFFL, …
&tname	AESTDTCx = AESTDTC; AETERMx = AETERM; AEYNx = AEYN; SAFFLx = SAFFL; …
&tnamex	AESTDTC = AESTDTCx; AETERM = AETERMx; AEYN = AEYNx; SAFFL = SAFFLx; …
&CVar	AESTDTC AETERM AEYN SAFFL SEX …
&CVarx	AESTDTCx AETERMx AEYNx SAFFLx SEXx …
&CVarLen	AESTDTC $50 AETERM $200 AEYN $1 SAFFL $1 …

Step 4. Find the Maximum Length Value for each Variable

```
DATA  VarLen ;
   SET  &inp.;
   &newvar;
RUN;

PROC SQL NOPRINT ;
   CREATE TABLE maxx  AS
   SELECT &maxvar
   FROM VarLen  ;
QUIT;
```

Step 5. Get Maximum Length Values

```
DATA maxx_t2;
      LENGTH  name6  $100;
      SET maxx_t ;
      IF col1<1 THEN col1=1;
      name6 = TRIM(name)||'  $'||STRIP(PUT(col1, best.));
RUN ;
```

Then create a macro variable to store the new length of each character variable.

```
PROC SQL  NOPRINT ;
   SELECT  strip(name6)  INTO: newlen   SEPARATED BY ' '
    FROM maxx_t2 ;
QUIT;
```

Step 6. Change Variable Names to Temporary Names

```
DATA  temp (drop = &CVar. ) ;
     LENGTH  &CVarLen  ;
     SET  &inp.;
     &tname.;
RUN;
```

Step 7. Change Temporary Names Back to Original Variable Names with New Lengths

```
DATA  newONE  (drop= &CVarx.) ;
      LENGTH &newlen.  ;
      SET  temp ;
      &tnamex.;
RUN;
```

Conclusion

The above seven steps are simple and straightforward. The macro can be used for minimizing the size of any SAS datasets so as to save disk space and improve process efficiency.

Reference

[1].http://support.sas.com/documentation/cdl/en/hostunx/61879/HTML/default/viewer.htm#a000344718.htm