PharmaSUG 2017 - Paper QT01

Remove the Error: Variable Length is Too Long for Actual Data

Eric Larson, inVentiv Health, Sun Prairie, WI

ABSTRACT

Have you ever seen the error from Pinnacle 21 output 'Variable length is too long for actual data'?

I have created a macro that will go through the character variables and identify the minimum length to hold the longest value of the data, and apply that length to the dataset just before creating a SAS® transport file from it.

INTRODUCTION

This paper will present an option for creating SAS transport files from SAS datasets that will shorten the length of the character variables to the minimal length needed to hold the longest value. There are two macros, %MKXPT and %TRIMDS. This paper will focus on %TRIMDS, which is called by %MKXPT, and does the work to shorten the character variables.

The %MKXPT macro takes three parameters

- SRCFLDR – Path to the SAS datasets being converted to SAS Transport files.
- OUTPATH – Path to the location of where the SAS Transport files are to be created.
    - An output folder named like DDMONYYYY will be created and a temporary working folder will be created in here.
- KEEPTMP – Boolean value (Y/N) to indicate to keep the interim datasets in the temporary folder after they have been shortened and before the SAS Transport file is created.

The %TRIMDS works on the datasets in SRCFLDR following these steps:

1. Read the Name and Type of the variables.
2. For character type variables, create SAS code to read the distinct lengths of the values.
3. Merge the length values of the variables into one dataset.
4. Create LENGTH statements and apply them to the current dataset.

READING THE NAMES AND TYPES OF THE VARIABLES IN THE DATASET

To read the names and types of the variables the code uses the SASHELP library in the following way:

```
PROC SORT
   DATA=sashelp.vcolumn(WHERE=(    libname="TRMDS"
                               and memname=upcase("&_combds")))
   OUT=trm_vars(KEEP=name type varnum)
   ;
   BY name;
RUN;
```

NAME	TYPE	VARNUM
STUDYID	char	1
DOMAIN	char	2
EXSEQ	num	3
EX---	< char | num >	<n>

Table 1. Example of TRM_VARS dataset.

The TRMDS library is setup in the %MKXPT macro and the macro variable _combds is the parameter that holds the name of the dataset. VARNUM is kept to maintain the ordering of the variables when the new lengths are applied.

GET THE DISTINCT LENGTHS FOR CHARACTER VARIABLES

To get the distinct lengths of the character variables, the %TRIMDS macro creates SAS code based on the values in the dataset trm_vars. To do this the macro creates a temporary FILEREF and then writes SAS code for each variable in the datasets being processed to calculate the maximum length value of the variables. The macro then runs the generated code and releases the temporary file.

The code generated is like:

```
PROC SQL;

   create table _minlen_1 as
   select distinct "EX" as combds
      ,   max(length( DOMAIN )) as DOMAIN
   from TRMDS.EX
   where not missing(<NAME>);

   create table _minlen_2 as
   select distinct "EX" as combds
      ,   8 as EXSEQ
   from TRMDS.EX
   where not missing( EXSEQ );

   create table _minlen_<n> as
   select distinct "<DATASET>" as combds
      ,   max(length(<NAME>)) as <NAME>
   from TRMDS.<DATASET>
   where not missing(<NAME>);

   ……

QUIT;
DATA trm_TEMP;
   MERGE _minlen_1 _minlen_2 … ;
   BY combds;
RUN;
```

The datasets that are created are similar to:

COMBDS	STUDYID	DOMAIN	EXSEQ	EX---
EX	22	2	8	<n>

Table 2. Example of dataset generated after finding the lengths of the variables.

Each dataset is then transposed by the variable COMBDS:

```
PROC TRANSPOSE
      DATA = trm_TEMP
      OUT  = trm_TEMP ( RENAME= ( _name_ = name
                                  col1   = newlength));
   BY combds;
RUN;
```

COMBDS	NAME	NEWLENGTH
EX	STUDYID	22
EX	DOMAIN	2
EX	EXSEQ	8
EX	EX---	<n>

Table 3. Example of TRM_TEMP dataset.

ADD THE LENGTH VALUES ONTO THE ORIGINAL DATASET

The datasets in Table 1 and in Table 3 are then merged together by the variable NAME and processed to produce the length statements needs to apply to the dataset identified by the macro variable _COMBDS. If a variable is a Character type a variable is added that holds a '$' and the NEWLENGTH variable is adjusted to be the maximum of 1 or the current value. If the value of NEWLENGTH is >200, a Warning is printed out to the LOG as "WARNING: Variable <DATASET>.<VARIABLE> has been truncated to 200 characters."

VARNUM	TRIMCMD
1	LENGTH STUDYID $ 22 ;
2	LENGTH DOMAIN $ 2 ;
3	LENGTH EXSEQ 8 ;
<n>	<generated length statement>

Table 4. Example of TRM_VARS dataset after merging and processing.

APPLY THE LENGTH STATEMENT TO THE DATASET BEING PROCESSED

The macro variable _NEWLEN is created from the variable TRIMCMD that is then used in a datastep to reset the lengths of the variables.

This variable is created by the following code:

```
PROC SQL NOPRINT;
   SELECT trimcmd INTO : _newlen SEPARATED BY ' '
   FROM trm_vars
   ;
QUIT;
```

Once this is done the length code is applied to the dataset:

```
PROC DATA TRMDS.&_combds.(label="&trm_dslbl.");
   &_newlen.
   SET TRMDS.&_combds.;
RUN;
```

As this code is applied the following warning will be produced for each variable that has a change in length: "WARNING: Multiple lengths were specified for the variable <NAME> by input data set(s). This can cause truncation of data."

This warning cannot be avoided, but can be "rerouted" by adding code to output the LOG to a new location and then restoring it to the original LOG

After this step, the transport file is made for this dataset, and the process is repeated for every dataset in the source folder.

CONCLUSION

These two macros will create transport files that will not produce the Pinnacle 21 error. In addition to removing the error, this is a way for programmers to leave the length of character variables at 200 during the course of a study, so as to eliminate the need of having to increase the variable length every time new data arrives while developing the code.

CONTACT INFORMATION

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

Eric Larson
inVentiv Health
ejl3021@gmail.com

APPENDIX 1: SAS MACRO CODE

```
/*************************************************************************
***
***  Program:      MKXPT.SAS
***  Programmer:   Eric Larson
***  Date Created: 2016-11-16 12:33:09 PM
***
***  Purpose:      Creates SAS XPT files after the character variables have
***                  been reduced in size according to FDA guidelines
***
***  Comments:     Uses the TRIMDS macro to reduce the length of character
***                REQUIRES the use of the x command
***
***                SRCFLDR is the folder path to the source data to be
***                  converted to XPT files
***
***                OUTPATH is the folder path for the output of the XPT files
***                  this folder will have a new folder created within  for
***                  the output
***
***  Software:     Runs under OS Windows on SAS version 9.3 or greater.
***
***  Modification History:
***
*** Date       Programmer          Description
*** ---------- ------------------- --------------
*** yyyy-mm-dd First and last name
***
*************************************************************************
*** Variable length is too long for actual data
*************************************************************************/
%macro mkxpt(
      srcfldr =
   ,  outpath =
   ,  keeptmp = N
   );

   proc datasets lib=work memtype=data kill;
   run;
   quit;

   *** Check for the source folder ***;
   %if %sysfunc(fileexist(&srcfldr.)) eq 0 %then %do;
      %put %str(ERR)OR: Source folder path does not exist. Ending MKXPT macro.;
      %goto _endmac;
   %end;

   *** Check for the output folder ***;
   %if %sysfunc(fileexist(&outpath.)) eq 0 %then %do;
      %put %str(ERR)OR: Output path does not exist. Ending MKXPT macro.;
      %goto _endmac;
   %end;

   *** set macro variable to hold a new fodler name ***;
   *** use a date value to use for the new folder name ***;
   %let DTVAL = &SYSDATE9;


   *** Check if the new folder exists ***;
   %if %sysfunc(fileexist(&outpath.\&DTVAL.)) eq 0 %then %do;
      *** if not, create the folder ***;
      %LET XPTlib = %SYSFUNC(DCREATE(&DTVAL., &outpath.));
```

```
%end;
%else %do;
   *** if so, set a macro variable to the folder ***;
   %LET XPTlib = &outpath.\&DTVAL. ;
%end;


*** A temporary folder needs to be created ***;
*** Check if the temporary folder exists ***;
%if %sysfunc(fileexist(&XPTlib.\trimmed)) eq 0 %then %do;
   *** if not, create the folder ***;
   %LET trmlib = %SYSFUNC(DCREATE(trimmed, &XPTlib.));
%end;
%else %do;
   *** if so, set a macro variable to the folder ***;
   %LET trmlib = &XPTlib.\trimmed ;
%end;

*** Create the libname references ***;
libname INDS "&srcfldr." access=readonly;
libname TRMDS "&trmlib." ;

*** remove the current datasets in the Temporary library ***;
proc datasets library=TRMDS memtype=data kill;
run;
quit;



*** Retrieve the names of the datasets in the source folder ***;
proc sql noprint;
   select distinct upcase(memname) into: CPDSETS separated by ' '
   from dictionary.tables
   where upcase(libname)="INDS"
   ;
quit;

*** Loop through each dataset ***;
%do i = 1 %to %sysfunc(countw(&CPDSETS., %str( )));
   %let currds = %scan(&CPDSETS., &i, %str( ));
   %put >>>>> &currds will be processed.;

   *** Store the dataset label ***;
   proc sql noprint;
      select strip(memlabel) into: trm_dslbl
      from sashelp.vtable
      where upcase(libname)="INDS" and memname=upcase("&currds")
      ;
   quit;
   %let trm_dslbl = %trim(%left(&trm_dslbl.));
   %put Copying &currds. (&trm_dslbl.) to working folder;

   *** Copy the dataset from the source folder to the ***;
   *** temporary working folder ***;
   proc copy in=INDS out=TRMDS memtype=data;
      select &currds;
   run;

   %put Trimming &currds. ;

   *** Run the Trimming macro on the current dataset ***;
   %trimds(&currds);
```

```
      %put Creating Transport file (&currds..xpt);

      *** Create the transport file for the current dataset ***;
      libname xpt xport "&XPTlib.\%lowcase(&currds.).xpt";
      proc copy in=TRMDS out=xpt memtype=data;
         select &currds;
      run;
      libname xpt clear;


   %end;

   *** Clear the library references ***;
   libname TRMDS clear;
   libname INDS clear;

   *** Remove the temporary folder and working datasets ***;
   %if &keeptmp = N %then %do;
      x "rd /s /q &trmlib.";
   %end;

   %_endmac:

%mend mkxpt;







/*************************************************************************
***
***  Program:      TRIMDS.SAS
***  Programmer:   Eric Larson
***  Date Created: 2016-11-16 12:40:10 PM
***
***  Purpose:      Reduces character lengths in accordance to FDA guidelines
***
***  Comments:     Libref TRMDS must be defined prior to using this macro
***                If a length is greater than 200 a warning will be produced
***                  indicating the dataset and variable that was truncated.
***
***  Software:     Runs under OS Windows on SAS version 9.3 or greater.
***
***  Modification History:
***
*** Date       Programmer          Description
*** ---------- ------------------- --------------
*** yyyy-mm-dd First and last name
***
*************************************************************************
***
*************************************************************************/
%macro trimds(_combds);


   *** retrive the NAME, TYPE, and Order of variables for the dataset ***;
   proc sort
      data=sashelp.vcolumn(where=(libname="TRMDS" and memname=upcase("&_combds")))
      out=trm_vars(keep=name type varnum)
      ;
```

```
      by name;
   run;


   ************************************************************************;
   *** This section parses the variable names of the dataset and creates SAS
   *** code that will get the minimum length needed to hold the longest data
   *** within the character variables
************************************************************************;
   filename TRIMTMP temp;
   data _null_;
   file TRIMTMP;
   set trm_vars end=eof; *(where=(upcase(type) = 'CHAR')) ;
         by name;
      if _n_ = 1 then do;
         put 'proc sql;';
      end;
      put 'create table _minlen_' _n_ ' as';
      put 'select  distinct "' "&_combds."  '" as combds ';
      if upcase(type) = 'CHAR' then do;
         put ", max(length(" NAME ")) as " NAME  ;
      end;
      if upcase(type) = 'NUM' then do;
         put ", 8 as " NAME  ;
      end;
      put 'from TRMDS.' "&_combds.";
      put 'where not missing(' NAME ')';
      put ';';

       if eof then do;
         put 'quit;';
         put 'data trm_TEMP;';
         put 'merge';
         do i = 1 to _n_;
            put '_minlen_' i ;
         end;
         put ';';
         put 'by combds;';
         put 'run;';
         put 'proc datasets nolist;';
         put 'delete _minlen_:;';
         put 'run; quit;';
      end;
   run;
   %include TRIMTMP;
   filename TRIMTMP clear;

   proc transpose data=trm_TEMP out=trm_TEMP(rename=(_name_=name col1=newlength));
      by combds;
   run;

   proc sort data= trm_TEMP(drop=combds);
      by name;
   run;

   *** merge the derived lengths onto the working dataset ***;
   data trm_vars;
      merge trm_vars
            trm_TEMP;
      by name;
   run;

   *** Parse the new length data and create 'length' code for each ***;
```

```
   *** variable in the current dataset ***;
   data trm_vars;
      set trm_vars;
      length TYPEind $1 trimcmd $200;
      if upcase(type) = 'CHAR' then do;
         TYPEind = "$";
         newlength = max(1,newlength);
      end;
      else do;
         call missing(TYPEind);
      end;
      if upcase(type) = 'CHAR' then do;
         if newlength > 200 then do;
            newlength = 200;
            put "%str(WARN)ING:  Variable &_combds.." NAME "has been truncated to 200
characters.";
         end;
      end;
      trimcmd = catx(" ","LENGTH",NAME,TYPEind,vvalue(newlength),";");
      keep varnum trimcmd;
   run;

   *** Arrange the variables in the original order ***;
   proc sort data= trm_vars;
      by varnum;
   run;

   *** Assign a macro variable that holds the new length codes ***;
   proc sql NOPRINT;
      select trimcmd into : _newlen separated by ' '
      from trm_vars
      ;
   quit;

   *** Apply the new lengths to the working dataset in the temporary folder ***;
   data TRMDS.&_combds.(label="&trm_dslbl.");
      &_newlen.
      set TRMDS.&_combds.;
   run;

%mend trimds;
```