PharmaSUG 2013 - Paper CC33

Macro to Conduct Consistency Checks

Walter H. Hufford, Jr., Novartis Pharmaceuticals Corporation, East Hanover, NJ

ABSTRACT

Pharmaceutical organizations are transitioning from legacy data standards to the CDISC Study Data Tabulations Model (SDTM) and the Analysis Data Model (ADaM) in anticipation of the FDA mandating their use in electronic submissions. Moving to these new data standards is fraught with challenges and there is usually a steep learning curve followed by a bumpy implementation. Although most Data Managers and Statistical Programmers working in the pharmaceutical industry have been exposed to these data models, most are not experts in them yet. This often leads to inconsistent data set and variable attributes - the nemesis of all Statistical Programmers. PROC COMPARE is a useful tool when comparing 2 datasets for inconsistencies; however, it does not provide the ability to compare more than 2 datasets simultaneously. This paper presents a generic macro designed to identify data set and variable attribute discrepancies across N datasets simultaneously and export those discrepancies into a user friendly Excel format for quick review and resolution.

INTRODUCTION

The macro presented in this paper uses several basic SAS® programming techniques to dynamically compare N datasets across N studies and produce a variable attribute discrepancy report. These include PARMBUFF, use of SAS dictionary files, CNTLIN to build a dynamic format, the ODS ExcelXP tagset and style elements in PROC REPORT. Two discrepancy report examples are provided. The first example compares variable attributes across four different studies. The second example compares variable attributes from three studies against a "gold-standard" or "target".

START OF MACRO

The only input required by the macro is a list of LIBREFS containing the datasets to compare for each study. The PARMBUFF option allows the macro to accept a varying number of LIBREFS. The code below displays the macro opening statement using the PARMBUFF option, a simple %PUT statement to feedback the value of the macro parameter to the LOG and a %LET statement which obtains the name of the first LIBREF from the list.

```
%macro StrucChk/parmbuff ;
 %put syspbuff contains: &syspbuff ;
 %let num=1 ;
 %let study=%scan(&syspbuff,&num) ;
```

MACRO PARAMETER VALIDATION

All macros should behave nicely even if users do not. This is easily achieved via default parameters and soft landings when macro execution halts due to user error. The only user defined parameter required for this macro contains a list of LIBREFs for each study to be compared so a default is not possible. The code below displays three parameter checks users are most likely to trip on. The macro quietly halts execution and provides a friendly (and hopefully useful) description of the user error should any of the rules be broken.

```
%let e = 0 ;
   %if &syspbuff= %then %do;
       %put ERROR:  You must supply a parameter to macro StrucChk. ;
       %let e = 1 ;
   %end;
   %let chk=1 ;
   %let chklib=%scan(&syspbuff,&chk) ;
    %do %while(&chklib ne) ;
      %if (%sysfunc(libref(&chklib))) %then %do ;
          %put ERROR:  Invalid LIBREF - &chklib does not exist ;
            %let e = 1 ;
      %end ;
      %let chk=%eval(&chk+1) ;
      %let chklib=%scan(&syspbuff,&chk) ;
   %end ;
```

```
%if %eval(&chk-1) < 2 %then %do ;
     %put ERROR:  Macro requires at least 2 LIBREFS to compare ;
        %let e = 1 ;
%end ;
%if &e = 1 %then %goto exit ;
%put NOTE:  StrucChk checks completed successfully. ;
```

DATASET ATTRIBUTES

PROC SQL is used to interrogate the SAS dictionary files and create temporary datasets containing variable attribute information for each study (e.g., STUDY1, STUDY2…). A simple DO LOOP rolls thru each of the studies provided to the macro. DO WHILE checks execution at the bottom of the loop so once the last study is processed no more looping occurs. The same task could have been accomplished using the SAS help files or PROC CONTENTS.

```
%do %while(&study ne) ;
   proc sql ;
     create table STUDY&num as
     select libname, memname, name, type, length, format, label
     from dictionary.columns
     where LIBNAME="%upcase(&study)"
     order by memname, name, type, length, libname ;
   quit ;
   %let study_name&num=%UPCASE(%scan(&syspbuff,&num)) ;
   %let num=%eval(&num+1) ;
   %let study=%scan(&syspbuff,&num) ;
%end ;
   %let num=%eval(&num-1) ;
```

IDENTIFY DISCREPANCIES

The temporary datasets containing variable attributes are merged together BY the variable attributes under comparison. The IN= values from the MERGE statement are used to create a series of binary numbers. These binary numbers are then used to form a base (2) representation (e.g., 1111) which is stored in the DIS_FLG variable in decimal format (see yellow highlighted code below). By back converting to the base (2) representation, the DIS_FLG variable is used to identify variable attribute discrepancies as well as the source contributors. Assuming four studies being compared, if all four have the same attributes for a variable DIS_FLG = 15 (e.g., 1111 = $2^3 + 2^2 + 2^1 + 2^0 = 15$); if only the first and third LIBREF have the same attributes for a variable DIS_FLG = 5 (e.g., 0101 = $2^2 + 2^0 = 5$).

The red highlighted code below performs a variable attribute consistency check against a “gold-standard”. By default the macro assigns the first study provided by the user as the “gold-standard” against which all other studies are compared.

```
data check ;
 merge %do I = 1 %to %eval(&num) ;
          STUDY&i (IN=in&i rename=(libname=lib&i ))
       %end ; ;
 by memname name type length format label ;
 array targ [&num] _temporary_ ( %do i = 0 %to %eval(&num - 1) ;
                                    %eval(2**&i)
                                 %end ; ) ;
 if _N_ = 1 then do ;
     gtot = sum(%do J=1 %to %eval(&num - 1);
                   targ[&j],
                %end;
                targ[%eval(&num)]) ;
     call symput("tot",Gtot) ;
 end ;
 dis_flg = 0 ;
 %do K = 1 %to %eval(&num) ;
     if in&k then dis_flg = dis_flg + targ(&k) ;
          if in1 and not in&k then do ;
               ds&k._lib = lib&k ;
          ds&k._dis_flg = 'Non-match with Standard' ;
```

```
            end ;
            else if not in1 and in&k then do ;
                 ds&k._lib = lib&k ;
            ds&k._dis_flg = '++++++++ Extra ++++++++' ;
            end ;
   %end ;
 run ;
```

DYNAMIC FORMAT

The objective of the macro is to identify variable attribute inconsistencies across studies and note them in a discrepancy file. For example, only study F2302s contains a variable named ADSL.AGEU or studies F2306s and F2309s contain a variable named ADSL.RACE without the $RACEF. format. The code below dynamically builds a format for the new DIS_FLG variable to associate the DIS_FLG variable value with a text description of the studies containing the unique variable attributes.

```
%local strt i a x ;
%let strt = 1 ;

data cntlin (keep=fmtname type start label) ;
  retain sp ' ' fmtname 'dis_cde' type 'N' ;

  %do i = 1 %to &num ;
     do var&i = &strt to &num ;
        %do z = 1 %to &num ;
            if var&i = &z then varb&i = %eval(2**%eval(&z-1)) ;
                 if var&i = &z then varbn&i = "&&study_name&z" ;
        %end ;
            start = sum(of varb1-varb&i) ;
        label = catx(sp,of varbn1-varbn&i) ;
        output ;
     %let strt = %str(%(var&i + 1 %)) ;
  %end ;
  %do i = 1 %to &num ;
     end ;
  %end ;
run ;

proc format cntlin=cntlin ;
run ;
```

DISCREPANCY REPORT

Finally, the code below produces the variable attribute discrepancy report. The ODS ExcelXP tagset is used to produce an XML file that may be opened in Excel. Style elements are incorporated into the PROC REPORT to produce traffic-lighting. This highlights rows requiring attention of the reviewer. Recall the example above of four studies being compared, all four have the same attributes for a variable so DIS_FLG = 15 (e.g., 1111 = $2^3 + 2^2 + 2^1 + 2^0$); if only the first and third LIBREF have the same attributes for a variable DIS_FLG = 5. For this example, the TOT macro variable resolves to 15. The COMPUTE block in the PROC REPORT is used to perform a CALL DEFINE statement containing style elements used to change the color of the row being displayed. If all studies have the same attribute for a variable then DIS_FLG = &TOT and the row color is set to light blue and identifies the row as a match not requiring further review or action. If only a single study has an attribute for a variable then the color of the row is set to red drawing reviewer attention. All other row colors are set to yellow since more than 1 but less than 4 studies have the variable attribute. In these examples yellow and red do not always indicate an issue but probably require review. Note that if all 4 studies define a variable attribute incorrectly then a blue row indicates a consistent approach but not necessarily a correct approach!

```
ods listing close ;

ODS tagsets.ExcelXP path="/vob/&irproju./&protno./report/pgm_a/"
                    file="PharmaSUG_Discrepancy_Traffic_Lighting_Example.xml"
                    style=Printer options(embedded_titles='yes'
                    embedded_footnotes='yes' print_header='&C&A&RPage &P of &N'
                    print_footer='&RPrinted &D at &T' autofilter='all'
```

```
                    orientation='landscape' autofit_height='yes'
                    absolute_column_width='18') ;

TITLE1 "Attribute Consistency Checks for Studies &syspbuff" ;
FOOTNOTE "Project:  SuperDrug123" ;

ODS tagsets.ExcelXP options(sheet_name='Example 1') ;

proc report data=check nowindows headline headskip ;
 column memname name type length label format dis_flg light ;
 define memname / display 'Member' ;
 define name    / display 'Variable' ;
 define type    / display 'Type' ;
 define length  / display 'Length' LEFT ;
 define format  / display 'Format' ;
 define label   / display 'Label' ;
 define dis_flg / display 'Members Contributing Attributes'
                  format=dis_cde. LEFT ;
 define light   / computed noprint ;

 compute light ;
   if dis_flg in(%do i = 0 %to %eval(&num - 1) ; %eval(2**&i) %end ;)then do ;
      call define(_row_,"style","style=[background=lightRED]") ;
   end ;
   else if dis_flg = &tot then do ;
      call define(_row_,"style","style=[background=lightBLUE]") ;
   end ;
   else do ;
      call define(_row_,"style","style=[background=YELLOW]") ;
   end ;
 endcomp ;
run ;
```

Attribute Consistency Checks for Studies (F2302s, F2305s, F2306s, F2309s)

Member	Variable	Type	Length	Label	Format	Members Contributing Attributes
ADSL	AGE	num	8	Age		F2302S F2305S F2306S F2309S
ADSL	AGEU	char	20	Age Units	$AGEU.	F2302S
ADSL	ARM	char	80	Description of Planned Arm		F2302S F2305S F2306S F2309S
ADSL	COMPLFL	char	20	Completers Population Flag		F2302S F2305S F2306S F2309S
ADSL	ENRLFL	char	20	Enrolled Population Flag		F2302S F2305S F2306S F2309S
ADSL	FASFL	char	20	Fulle Analysis Set Population Flag		F2302S F2305S F2306S F2309S
ADSL	ITT	char	20	Intent-To-Treat Population Flag		F2302S
ADSL	ITTFL	char	20	Intent-To-Treat Population Flag		F2302S
ADSL	PPROTFL	char	20	Per-Protocol Population Flag		F2302S F2305S F2306S F2309S
ADSL	RACE	char	20	Race		F2306S F2309S
ADSL	RACE	char	20	Race	$RACE.	F2302S F2305S
ADSL	RACEGR1	char	80	Pooled Race Group 1		F2302S F2305S F2306S F2309S
ADSL	RACEGR1N	num	8	Pooled Race group 1 (N)		F2302S F2305S F2306S F2309S
ADSL	RANDFL	char	20	Randomized Population Flag		F2302S F2305S F2306S F2309S
ADSL	SAFFL	char	20	Safety Population Flag		F2302S F2305S F2306S F2309S
ADSL	SEX	char	20	Sex	$SEX.	F2302S F2305S F2306S F2309S
ADSL	SITEGR1	char	40	Pooled Site Group 1		F2302S F2305S F2306S F2309S
ADSL	SITEGR1N	num	8	Pooled Site Group 1 (N)		F2302S F2305S F2306S F2309S
ADSL	SITEID	char	20	Study Site Identifier		F2302S F2305S F2306S F2309S
ADSL	STUDYID	char	40	Study Identifier		F2302S F2305S F2306S F2309S
ADSL	SUBJID	char	40	Subject Identifier for the Study		F2302S F2305S F2306S F2309S
ADSL	TPT1	char	80	Planned Treatment for Period 2		F2306S
ADSL	TRT01P	char	80	Planned Treatment for Period 1		F2302S F2305S F2306S F2309S
ADSL	TRT01PN	num	8	Planned Treatment for Period 1 (N)		F2302S F2305S F2306S F2309S
ADSL	TRT02P	char	80	Planned Treatment for Period 2		F2302S
ADSL	TRT02PN	num	8	Planned Treatment for Period 2 (N)		F2302S
ADSL	TRT1PN	num	8	Planned Treatment for Period 2 (N)		F2306S
ADSL	USUBJID	char	40			F2305S
ADSL	USUBJID	char	40	Unique Subject Identifier		F2302S F2306S F2309S

Project: SuperDrug123

Display 1. Variable Attribute Discrepancy Report

DISCREPANCY REPORT USING A "GOLD-STANDARD" OR TARGET

The previous example compares four studies none of which are considered the "gold-standard" or "target" which all studies must adhere to. The PROC REPORT below creates a discrepancy report using the first study provided by the user as the "gold-standard" against which all other studies are compared. Once again style elements are incorporated into the PROC REPORT to produce traffic-lighting. However, this example uses a style element on the DEFINE statement of the PROC REPORT to change the color of (e.g., highlight) individual cells rather than entire rows. The CALL DEFINE within the COMPUTE block of the PROC REPORT is used to change the color of every other row to facilitate user review across a row.

```
ODS tagsets.ExcelXP options(sheet_name='Example 2');

proc format ;
 value $sigb 'Non-match with Standard' = 'RED'
             '++++++++ Extra ++++++++' = 'BLUE' ;
run ;

proc report data=check nowindows headline headskip ;
 column memname name type length label format
        ds2_dis_flg ds3_dis_flg ds4_dis_flg light ;
 define memname     / display ;
 define name        / display ;
 define type        / display 'Type' ;
 define length      / display 'Length' LEFT ;
 define format      / display 'Format' ;
 define label       / display 'Label' ;
 define light       / computed noprint ;
 define ds2_dis_flg / display 'F2305s'
                      style={font_weight=bold foreground=$sigb.} ;
 define ds3_dis_flg / display 'F2306s'
                      style={font_weight=bold foreground=$sigb.} ;
 define ds4_dis_flg / display 'F2309s'
                      style={font_weight=bold foreground=$sigb.} ;

 compute light ;
   line_count + 1 ;
   if mod(line_count,2)=0 then do ;
      call define(_row_,"style","style=[background=lightYELLOW]") ;
   end ;
 endcomp ;

run ;

ODS tagsets.ExcelXP CLOSE ;
```

Attribute Consistency Checks for Studies (F2302s, F2305s, F2306s, F2309s)

Member	Column Nam	Type	Length	Label	Format	F2305s	F2306s	F2309s
ADSL	AGE	num	8	Age				
ADSL	AGEU	char	20	Age Units	$AGEU.	Non-match with Standard	Non-match with Standard	Non-match with Standard
ADSL	ARM	char	80	Description of				
ADSL	COMPLFL	char	20	Completers				
ADSL	ENRLFL	char	20	Enrolled				
ADSL	FASFL	char	20	Fulle Analysis				
ADSL	ITT	char	20	Intent-To-Treat		Non-match with Standard	Non-match with Standard	Non-match with Standard
ADSL	ITTFL	char	20	Intent-To-Treat		Non-match with Standard	Non-match with Standard	Non-match with Standard
ADSL	PPROTFL	char	20	Per-Protocol				
ADSL	RACE	char	20	Race			++++++++ Extra ++++++++	++++++++ Extra ++++++++
ADSL	RACE	char	20	Race	$RACE.		Non-match with Standard	Non-match with Standard
ADSL	RACEGR1	char	80	Pooled Race				
ADSL	RACEGR1N	num	8	Pooled Race				
ADSL	RANDFL	char	20	Randomized				
ADSL	SAFFL	char	20	Safety				
ADSL	SEX	char	20	Sex	$SEX.			
ADSL	SITEGR1	char	40	Pooled Site				
ADSL	SITEGR1N	num	8	Pooled Site				
ADSL	SITEID	char	20	Study Site				
ADSL	STUDYID	char	40	Study Identifier				
ADSL	SUBJID	char	40	Subject Identifier				
ADSL	TPT1	char	80	Planned			++++++++ Extra ++++++++	
ADSL	TRT01P	char	80	Planned				
ADSL	TRT01PN	num	8	Planned				
ADSL	TRT02P	char	80	Planned		Non-match with Standard	Non-match with Standard	Non-match with Standard
ADSL	TRT02PN	num	8	Planned		Non-match with Standard	Non-match with Standard	Non-match with Standard
ADSL	TRT1PN	num	8	Planned			++++++++ Extra ++++++++	
ADSL	USUBJID	char	40			++++++++ Extra ++++++++		
ADSL	USUBJID	char	40	Unique Subject		Non-match with Standard		

Project: SuperDrug123

Display 2. Variable Attribute Discrepancy Report Using a "Gold-Standard"

CONCLUSION

The concepts contained in the macro presented in this paper might be considered as an alternative solution to PROC COMPARE when comparing variable attributes across more than two studies simultaneously. In addition, it could be used by SAS Programmers during development of standard datasets (e.g., SDTM, ADaM, corporate standard) to ensure they are "on-target". The code used to derive the DIS_FLG variable values could also be implemented in other applications that require identification of contributing data such as in oncology studies where several datasets are interrogated to identify date of death (e.g., ECOG, Subject Summary, Grade 5 Adverse Events).

REFERENCES

Karp, Andrew H., 2009. "Traffic-Lighting Your Reports the Easy Way with PROC REPORT and ODS." *SAS Global Forum 2011*. Cary, NC: SAS Institute, Inc.

Available at http://support.sas.com/resources/papers/proceedings11/TOC.html.

ACKNOWLEDGMENTS

I would like to thank John Wenston, Christina Scienski and Gregory Ligozio for their review and helpful suggestions.

CONTACT INFORMATION

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

Name: Walter H. Hufford, Jr.
Enterprise: Novartis Pharmaceuticals Corporation
Address: One Health Plaza
City, State ZIP: East Hanover, NJ 07936-1080
E-mail: Walter.Hufford@Novartis.com
Web: www.Novartis.com