

PhUSE 2010

Paper CC07

Generate ATTRIB and FORMAT Statements from the SAP

Shafi Chowdhury, Shafi Consultancy Limited, London, U.K.

ABSTRACT

Analysis datasets are often defined either in the Statistical Analysis Plan (SAP) or in associated specifications. An analysis dataset specification contains a list of all variables expected in the dataset together with their attributes, definitions and associated formats in the form of codes and decodes which may be required to create or display those variables. These are generally listed in two separate tables within the specification. All the variables, attributes and formats are then re-entered in the SAS[®] programs that creates the analysis datasets. This paper will provide macros which can be used to programmatically transfer these details to the analysis dataset program and avoid re-entering them manually. One macro will list all the variables in an ATTRIB statement which can be copied and pasted at the top of the final dataset to ensure all expected variables are present and have the exact attribute as defined in the specification. The other macro will create a FORMAT statement which can be placed either at the top of the program or in an AUTOEXEC.SAS. This will ensure the variables contained in the final dataset have the attributes and use the formats define in the final SAP.

INTRODUCTION

The attributes and formats defined in an analysis dataset specification have to be transferred to SAS in order to create the analysis dataset. They are usually re-entered in the SAS program by the programmer. This has all the risks of capturing the same data twice, including typo, out of date data and having either more or less variables than defined in the final specification. It can also be very time consuming if the dataset has a horizontal structure, which can have more than a hundred variables.

The process to remove manually entering this data in the SAS program requires copying the tables from the SAP, pasting it into EXCEL and then saving the file as a .CSV file. The macro will then read in the file and either create the ATTRIB statement or the FORMAT statement in the LOG window, depending on the macro selected. This can then be copied to the main part of the analysis dataset program, saving ourselves a great deal of time and ensuring the accuracy of the attributes and formats.

GENERATING THE ATTRIB STATEMENT

TABLE IN THE SPECIFICATION (WORD DOCUMENT)

The SAP should contain a table of the following structure, listing all the variables and their attributes:

Variable	Label	Data Type	Length	Format	Example	Comments
STUDY	Trial number	Char	9	\$9.	1000_0001	Study number as it appears in the raw data
PTNO	Patient number	Num	8	10.	1201	Patient number as it appears in the raw data
INVSITE	Site	Char	10	\$10.	SITE1201	Site Code
POPU	Population	Char	16	\$16.	FAS	Patient set being analysed (definition in the core SAP): FAS=Full Analysis Set
POPUNY	Patient in the population	Num	8	Yesnofmt.	1	1=Yes, 0=No

PhUSE 2010

COPY THE TABLE FROM WORD INTO EXCEL AND SAVE IT AS A CSV FILE

Its important to ensure that there are no carriage returns in the WORD table, so that when the table is copied to EXCEL then each row in the WORD table results in just one row in the EXCEL file. The CSV file will look no different when opened in EXCEL, however, in NOTEPAD it will look appear as shown below:

Variable	Label	Data Type	Length	Format	Example	Comments
STUDY	Trial number	Char,9	\$9.	1000_0001	Study number as it appears in the raw data	
PTNO	Patient number	Num,8,10,1201			Patient number as it appears in the raw data	
INVSITE	Site	Char,10	\$10.	SITE1201	Site Code	
POPU	Population	Char,16	\$16.	FAS	Patient set being analysed (definition in the core SAP): FAS=Full Analysis Set	
POPUNY	Patient in the population	Num,8	Yesnofmt.,1	"1=Yes, 0=No"		

PROGRAM TO CREATE THE ATTRIB STATEMENT

```
*****;
*** Create an ATTRIB statement using the CSV file created from ***;
*** the analysis dataset description in the SAP. ***;
*** ***;
*** NOTE: eof is referenced twice in case the last record is ***;
*** deleted due to missing variable label. ***;
*** ***;
*** a. Define positions where the VARIABLE name, FORMAT and ***;
*** LABEL should be positioned. ***;
*** b. On the first observation write ATTRIB in the LOG. ***;
*** If its the last observation then write the semi-colon ***;
*** to mark the end of the ATTRIB statement. ***;
*** c. Only keep observations if the variable has a label. ***;
*** d. If formats do not have a '.', then add a '.' at the end ***;
*** e. Flag variables with labels longer than 40 characters as ***;
*** they may need to be shortened. ***;
*** Write the VARIABLE FORMAT = ... LABEL = ".." statement ***;
*** ***;
*****;

%MACRO x_attrib(PATH=, CSV_FILE=);

%IF %LENGTH(&path.)=0 %THEN %LET path=%SYSFUNC(PATHNAME(pgm));;

DATA x_attrib;
  LENGTH varname $10. varlbl $80. datatype $4. varlen 8. varfmt $10.;
  INFILE "&path.\&csv_file." DSD DLM=',' LRECL=1000 PAD FIRSTOBS=2 MISSEVER
    END=eof;
  INPUT varname $ varlbl $ datatype $ varlen varfmt $;

%*a*;
  vnam=10;
  vfmt=23;
  vlbl=45;

%*b*;
  IF _N_=1 THEN PUT ' ATTRIB' @@;
  IF eof AND varlbl = ' ' THEN PUT @vnam ' ';

%*c*;
  IF varlbl NE ' ';

%*d*;
  varname = LOWCASE(varname);
```

PhUSE 2010

```

IF varfmt NE '' AND INDEX(varfmt, '.')=0 THEN varfmt=TRIM(varfmt)||'.';

%*e*;
  lblen=LENGTH(varlbl);
  IF lblen GT 40 THEN PUT @3 '*' lblen @@;

  PUT @vnam varname @vfmt 'FORMAT = ' varfmt @vlbl 'LABEL = "' varlbl + (-1)
  ''';

  IF eof THEN PUT @vnam ''';
RUN;

%MEND x_attrib;

```

EXAMPLE MACRO CALL AND THE OUTPUT IN THE LOG WINDOW

```

%x_attrib(PATH = C:\Secure_data\PhUse_2010_Berlin,
          CSV_FILE = attrib.csv);

```

Result in the LOG window:

```

ATTRIB study      FORMAT = $9.      LABEL = "Trial number"
      ptno        FORMAT = 10.      LABEL = "Patient number"
      invsite     FORMAT = $10.     LABEL = "Site"
      popu        FORMAT = $16.     LABEL = "Population"
      popuny      FORMAT = Yesnofmt. LABEL = "Patient in the population"
      ;

```

GENERATING THE FORMAT STATEMENT

TABLE IN THE SPECIFICATION (WORD DOCUMENT)

The SAP should contain a table of the following structure, listing all the format names, codes and decodes:

Format name	Code	Decode
AGEGRPDC	1	<=50 years
	2	>50 - <=60 years
	3	>60 - <=70 years
	4	>70 - <=80 years
	5	>80 years
ALCCDDC	0	Non drinker
	1	Avg. consumption
	2	Exc. consumption
POPUDC	RAND	Randomised set
	TS	Treated set
	FAS	Full analysis set

PhUSE 2010

Format name	Code	Decode
RACEADC	1	White
	2	Black
	3	Asian
SEXDC	1	Male
	2	Female
SMOKCDDC	0	Never smoked
	1	Ex-smoker
	2	Currently smokes
YESNOFMT	0	No
	1	Yes

COPY THE TABLE FROM WORD INTO EXCEL AND SAVE IT AS A CSV FILE

Its important to ensure that there are no carriage returns in the WORD table, so that when the table is copied to EXCEL then each row in the WORD table results in just one row in the EXCEL file. The CSV file will look no different when opened in EXCEL, however, in NOTEPAD it will look appear as shown below:

```
Format name,Code,Decode
AGEGRPDC,1,<=50 years
,2,>50 - <=60 years
,3,>60 - <=70 years
,4,>70 - <=80 years
,5,>80 years
ALCCDDC,0,Non drinker
,1,Avg. consumption
,2,Exc. consumption
POPUDC,ENROL,Enrolled set
,RAND,Randomised set
,TS,Treated set
,FAS,Full analysis set
RACEADC,1,White
,2,Black
,3,Asian
SEXDC,1,Male
,2,Female
SMOKCDDC,0,Never smoked
,1,Ex-smoker
,2,Currently smokes
YESNOFMT,0,No
,1,Yes
```

PROGRAM TO CREATE THE FORMAT STATEMENT

```
*****;
*** Create an ATTRIB statement using the CSV file created from ***;
*** the analysis dataset description in the SAP. ***;
*** ***;
```

PhUSE 2010

```
*** NOTE: eof is referenced twice in case the last record is ***;
***         deleted due to missing decode value. ***;
*** ***;
*** a. Write PROC FORMAT if its the first record and RUN if ***;
***     its the last record and there is no decode to print. ***;
*** b. Only keep observations with a decode value. ***;
*** c. The format name should only appear on the first line of ***;
***     codes for each format. ***;
***     Identify if its a character or a numeric format then ***;
***     create the appropriate VALUE statement. i.e. with a $ ***;
***     in front of the name if its a character format. ***;
*** d. Write out the CODE = DECODE part of PROC FORMAT. ***;
*****;
```

```
%MACRO x_format(PATH=, CSV_FILE=);
```

```
%IF %LENGTH(&path.)=0 %THEN %LET path=%SYSFUNC(PATHNAME(pgm));;
```

```
DATA x_format;
```

```
LENGTH fmtname $8. code $40. decode $80. fmttype $4;
```

```
INFILE "&path.\&csv_file." DSD DLM=', ' LRECL=1000 PAD FIRSTOBS=2 MISSEVER  
END=eof;
```

```
INPUT fmtname $ code $ decode $;
```

```
RETAIN fmttype;
```

```
%*a*;
```

```
IF _N_=1 THEN PUT 'PROC FORMAT;';
```

```
IF eof AND decode = '' THEN PUT @5 ' ';  
/ 'RUN;';
```

```
%*b*;
```

```
IF decode NE '';
```

```
fmtname = LOWCASE(fmtname);
```

```
%*c*;
```

```
IF fmtname NE '' THEN DO;
```

```
nooffmt+1;
```

```
IF nooffmt GT 1 THEN PUT @5 ' ';;
```

```
IF INPUT(code,?? BEST.) NE . THEN DO;
```

```
fmttype = 'num';
```

```
PUT ' VALUE ' fmtname;
```

```
END;
```

```
ELSE DO;
```

```
fmttype = 'Char';
```

```
PUT ' VALUE $' fmtname;
```

```
END;
```

```
END;
```

```
%*d*;
```

```
IF fmttype = 'Char'
```

```
THEN PUT @5 ' "' code + (-1) ' "' @17 ' = "' decode + (-1) ' "';
```

```
ELSE PUT @5 code @7 ' = "' decode + (-1) ' "';
```

```
IF eof THEN PUT @5 ' ';
```

```
/ 'RUN;';
```

```
RUN;
```

```
%MEND x_format;
```

PhUSE 2010

EXAMPLE MACRO CALL AND THE OUTPUT IN THE LOG WINDOW

```
%x_format(PATH = C:\Secure_data\PhUse_2010_Berlin,  
          CSV_FILE = formats.csv);
```

Result in the LOG window:

```
PROC FORMAT;  
  VALUE agegrpdc  
    1 = "<=50 years"  
    2 = ">50 - <=60 years"  
    3 = ">60 - <=70 years"  
    4 = ">70 - <=80 years"  
    5 = ">80 years"  
  ;  
  VALUE alccddc  
    0 = "Non drinker"  
    1 = "Avg. consumption"  
    2 = "Exc. consumption"  
  ;  
  VALUE $popudc  
    "ENROL" = "Enrolled set"  
    "RAND" = "Randomised set"  
    "TS" = "Treated set"  
    "FAS" = "Full analysis set"  
  ;  
  VALUE raceadc  
    1 = "White"  
    2 = "Black"  
    3 = "Asian"  
  ;  
  VALUE sexdc  
    1 = "Male"  
    2 = "Female"  
  ;  
  VALUE smokcddc  
    0 = "Never smoked"  
    1 = "Ex-smoker"  
    2 = "Currently smokes"  
  ;  
  VALUE yesnofmt  
    0 = "No"  
    1 = "Yes"  
  ;  
RUN;
```

CONCLUSION

This is a quick and easy way to copy variable names, attributes and formats from the analysis datasets specifications to the analysis dataset programs. This is a process that has to be performed in order to create SAS analysis datasets, and anything which can save time and reduce the risk of inconsistency between the specification and the final dataset will be of great benefit.

Analysis datasets consist of many variables, sometimes over a hundred, and entering these manually and checking that they match the specification is a very time consuming task. The same applies to formats defined in the specification and used to create and display variables. As CDISC structure is implemented throughout the pharmaceutical industry, the need to ensure variable names, attributes and formats are strictly observed will be even

PhUSE 2010

more important. These macros therefore will reduce the risk of inconsistencies between the SAP and the final analysis dataset, help to maintain standards and save time.

CONTACT INFORMATION

Shafi Chowdhury M.Sc.
Shafi Consultancy Limited
7 Blossom Way
Uxbridge
Middlesex, UB10 9LL, UK
Work Phone: +44 770 288 7219
Email: Shafi@shaficonsultancy.com
Web: www.shaficonsultancy.com

Brand and product names are trademarks of their respective companies.