PharmaSUG 2012 – Paper CC26

The power of using options COMPLETETYPES and PRELOADFMT

Naren Surampalli, Novartis Pharmaceuticals, Florham Park, NJ

ABSTRACT

As a statistical programmer in pharmaceutical industry, our job revolves around generating reports, which involve calculating counts and percentages across different treatments. To generate counts, most of us use procedures like frequency, sql etc., but these procedure doesn't display a missing group, if all the values of a particular group are missing in a data set. To display zeros for a missing group or treatment in our reports, we end up writing additional code, which is not very efficient. This paper will discuss a better way to count, handle missing values efficiently and show how to create totals and sub-totals easily using one or two steps.

INTRODUCTION

When we create table reports, most of the time we encounter a missing category or a missing treatment in our data. To handle these missing categories, we end up writing additional code, which is not very efficient and takes more time to program. To handle missing categories or groups efficiently, we can use 2 procedures format and means with options completetypes and preloadfmt. Using these options help us to display missing values with zeros and display a missing treatment or group in the table output.

Procedure means is mostly used to summarize data, but in this paper, we explore the proc means options like completetypes, preloadfmt, mlf and nway to show an efficient way to handle missing values in any category or create total columns or sub-total columns in 1 step.

Handling Missing values

In any data set, if a category is missing, we display that category with zeros in the output. To display zeros, we usually hard code the zeros for the missing category. In the below example, method 1 is the inefficient method and method 2 is the efficient method.

Example 1:

Consider demographic data, where there are no females, but our output requires us to display female category along with males. To achieve this we usually end up creating dummy categories and hard code zeros for this particular group. These steps usually involve more programming and sometimes end up modifying the code several times in a study. Below are 2 methods. The first method describes displaying females using proc freq and hardcoding. The second method is the efficient method using proc means with options completetypes and preloadfmt.

Method 1:

```
PROC FORMAT; /** create 2 formats trt for treatment and gen for gender **/
  VALUE trt
    1 = 'Active'
    2 = 'Placebo';
  VALUE gen
    1 = 'Male'
    2 = 'Female';
RUN;

/** the demog data set below shows test data with 2 treatments and only 1 gender
    information **/


DATA demog;
  INPUT subj $ treat gender;
  CARDS;
   101 1  1
   102 2  1
   103 1  1
```

```
   104 2  1
   105 1  1
   106 2  1
  ;
RUN;

 *** using proc freq to get missing gender;

PROC FREQ DATA = demog NOPRINT;
  TABLES gender*treat/out=gender (drop=percent);
RUN;

/** creating a data set to display all combination of

    treatment and gender values **/

DATA mis_gen (DROP=i j);
  DO j=1 TO 2;
    DO i=1 TO 2;
      treat = j;
      gender = i;
      count = 0;
      OUTPUT;
    END;
  END;
RUN;


PROC SORT DATA = mis_gen;
  BY treat gender;
RUN;

PROC SORT DATA = gender;
  BY treat gender;
RUN;

DATA gender; /** merging the both data sets gives the female rows in the output **/
  MERGE mis_gen(in=i) gender(in=j);
    BY treat gender;
    FORMAT treat trt. gender gen.;
 RUN;


PROC PRINT DATA = gender;
  FORMAT treat trt. gender gen.;
RUN;
```

Method 2:

```
 *** using proc means with options preloadfmt and completetypes;

DATA demog;
   SET demog;
     _count=1; *** create a temporary count variable used in proc means below.;
RUN;

PROC MEANS DATA = demog completetypes nway NOPRINT;
  FORMAT treat trt. gender gen.;
  CLASS treat gender/preloadfmt;
  VAR _count;
  OUTPUT OUT = gender1 (drop=_ :)  N=count;
RUN;
```

Category	Active	Placebo
Gender		
Male	3	3
Female	0	0

Table 1: Demographics for Gender (Method 1 or Method 2).

By using the options completetypes, we get all possible combination of class values. Preloadfmt loads formats based on the class variables. Please note that preloadfmt should always be used with completetypes. Nway is used to give the highest combination of values. In method 2, we get females information with the gender1 data set with just 1 step. Table 1 is generated using method 1 or method 2. The missing female information is displayed in the output. We can clearly see from the above 2 methods that method 2 is the most efficient method to display missing values and categories. In method 2, we use the options completetypes, nway and preloadfmt to get the desired result in 1 step. Please note that we need to create a flag called _count and assign it to 1 to be used in proc means procedure.

Creating Totals and Sub-Total columns:

Most of the reports in a clinical study involve more than 1 treatment and we definitely present total column and some times sub-total columns in the reports. To create a total column, we usually end up duplicating the data and create a dummy treatment. Sometimes if the data set is large like a lab data, duplicating the data will require more system resources and time.

Example 2:
This example discusses a clinical study with 3 treatment groups, where we want to summarize the race categories for each treatment groups and create total and sub-total columns. Method 1 creates total and sub-total columns using duplication of data and involving many programming steps. Method 2 shows creating total and sub-totals using proc means with options completetypes, mlf, nway and preloadfmt in a very efficient manner.

Method 1:

```
PROC FORMAT; /*** creating a format for race category **/
  VALUE rce
    1 = 'asian'
    2 = 'black';
  VALUE trt
    1 = "ACT25"
    2 = "ACT50"
    3 = "ACT75"
    4 = "Total(Act25 + Act75)"
    5 = "Total";
RUN ;

***** get the data with 2 race categories and 3 treatments.;

DATA ident;
  LENGTH race $5;
    INPUT subjid trtan race $;
    CARDS;
      101 1 asian
      102 2 black
      103 3 asian
      104 1 asian
      105 2 black
      106 3 asian
      107 1 black
   ;
 RUN;

/** to create total and sub-total columns, we duplicate data and assign new value to
    treatment **/
```

```
DATA subj;
  SET ident;
    OUTPUT;
    IF trtan IN (1,3) THEN DO;
      Trtan = 4;
      OUTPUT; /** duplicating data **/
    END;
      Trtan = 5; /** duplicating data **/
      OUTPUT;
RUN;

/*** creating all combination of treatments and race categories ***/

DATA dummy (drop=i j); LENGTH race $5;
  DO i=1 TO 5;
    DO j=1 TO 2;
      trtan=i;
      race=put(j,rce.);
      count=0;    *** hard coding 0 in method 1;
      OUTPUT;
    END;
  END;
RUN;

PROC SQL;
  create table race as select count(distinct subjid) as count,
    trtan, race from subj
    group by trtan, race
    order by trtan, race;
QUIT;

PROC SORT DATA = dummy;
  BY trtan race;
RUN;

/** merging both data sets will give all combination of treatments and race
    categories **/

DATA race;
  MERGE dummy(in=i) race(in=j);
    BY trtan race;
RUN;
```

Method 2:

```
/** creating a format for treatment. Defining a format is a must when using options
    completetypes and preloadfmt. **/

PROC FORMAT;
  VALUE drugfmt (multilabel)
    1 =  "ACT25"
    2 =  "ACT50"
    3 =  "ACT75"
    1,3 =  "Total(Act25 + Act75)"
    1,2,3 =  "Total"
    ;
RUN;


DATA ident;
  SET ident;
    _count=1; /** creating a temporary variable used in proc means **/
RUN;
```

```
PROC MEANS DATA=ident completetypes nway NOPRINT;
  VAR _count;
  CLASS trtan/mlf preloadfmt;
  CLASS race;
  format trtan drugfmt.;
  OUTPUT out=race1(drop=_:) N=count;
RUN;
```

Category	ACT25	ACT50	ACT75	Total	Total (ACT25 + CT75)
Race					
Asian	2	0	2	4	4
Black	1	2	0	3	1

Table 2: Demographics for Race (Method 1 or Method 2)

In method 2, we use the option mlf, which is an abbreviation for multi-label formats. If we use mlf, we have to specify the keyword multi-label in proc format. Mlf enables proc means to use multi-label formats. In the above example, we have created 2 new columns “Total” and “Total (Act25 + Act75)” easily by just updating proc format and calling means procedure.

Table 2 is generated using method 1 or method 2. It produces 2 new columns “Total” and “Total (Act25 + Act75)”. You can clearly see that method 2 has only 1 step compared to several steps with method 1. Using the options mlf, preloadfmt, nway and completetypes, we can easily create totals and sub-totals.

CONCLUSION

Utilizing options like completetypes, preloadfmt, mlf, and nway in proc means clearly shows that it’s a better way of handling missing values or missing groups. Using these powerful options helps us to avoid writing additional code, which otherwise takes lot of resources and time.

CONTACT INFORMATION

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

Name: Naren Surampalli
Enterprise: Novartis Pharmaceuticals
Address: 180 park avenue
City, State ZIP: Florham Park, NJ 08356
Work Phone: 862-778-0093
E-mail: narensurampalli@yahoo.com