Paper 185-27

Macros: Data Listings With Power

Daphne Ewing, Synteract, Inc., Ambler, PA

ABSTRACT

Displaying a variety of data in a consistent manner can be tedious and frustrating. Making adjustments to these displays, such as adding/moving/removing columns, can be time consuming and just as frustrating.

This paper will show the user how to set up macros which work together to make data listings consistent, easy to follow, and easy to modify. I will define several SAS macros, show how they interact with each other and the data, and give examples of input and output. Given these macros, I will also show how to modify displays easily.

INTRODUCTION

When preparing clinical data listings for the FDA, or other review boards, accuracy and consistency are essential. Implementing consistency can be a challenge, especially when there are multiple people working on the same project. This paper will show how to use several macros in a DATA _NULL_ step that work together to gain more consistent output. These macros also make changing data listings easy and fast.

REVIEW THE REQUIREMENTS

The first step is to have a layout of what data is to be presented for all data listings within a given project. These layouts can then be reviewed to identify the portions of the displays that do not change and those that do change. Usually, the heading at the top of each page will not change. One other portion that may be consistent across listings is the patient identifying information and the order in which it is displayed. The portions of the displays that change are usually the different data items on each display (i.e. Demography fields versus Adverse Event fields). The remainder of this paper will show a breakdown of the different portions of the data listing and the macros associated with them.

THE HEADING (HDG MACRO)

The heading of data listings can be comprised of items such as the name of the company, the protocol, the indication, the page number, the system date and time, the type of report (clinical or statistical), and possibly the appendix number. To generate the heading for all listings, a heading macro can be written to keep that part of the display consistent. Below is an example of a heading macro:

```
%macro HDG(HID,
           PAGE=PNUM,
           TITLE=TITLE,
           FNCT=FNCT);

  put @1 'PhunnyPharm, Inc.'

  %if "&PAGE" ne ""
     %then @%eval(&LS-25) 'PAGE ' &PAGE

   /  @1 'Protocol No. ABC-123'
   /  @1 'Clinical/Statistical Report'
      @%eval(&LS-36)
          "&PGM &SYSDATE &SYSTIME";

  %if "&FNCT" ne ""
     %then &FNCT + 3%str(;);

  %if "&TITLE" ne "" %then %do;
     &TITLE = put("&HID",$TABLE.)%str(;);
     put %CTRPUT(&TITLE) /;
  %end;
%mend;
```

This HDG macro is comprised of the PUT statements necessary to display the heading information for all tables. In the example above, the name of the company and page number appear on the first line. There is reference here to the LS macro variable. This is a global macro variable which is set prior to the DATA _NULL_ which defines the number of spaces to a line, or linesize. The protocol appears on the second line, and on the third line is the type of the report and the program name that created this report (&PGM), the system date and the system time. The two lines incrementing the FNCT macro variable are used to account for the number of lines necessary to display the default lines from the FOOT macro (which will be described in more detail later). The FNCT variable is used and incremented to handle the number of lines required of the footnote lines on each listing. Since, as shown below, the FOOT macro automatically displays a solid line, a blank space, and one line of text, this HDG macro adds 3 to the FNCT macro variable to handle the FOOT macro. This makes is possible to determine the number of actual data lines which can still be displayed on the table. The last portion of this macro is the displaying of the appendix number which is described in more detail below.

Since appendix numbers change fairly frequently, it is easiest to change them in one place, rather than editing every single program where there is an appendix number change. Every program for a given project creates a different display or group of displays. Each of which will have a different appendix number. Hence the PROC FORMAT is a solution to this problem. The following $TABLE format is an example:

```
proc format;
  value $TABLE
  'L.DEMO'   = 'Appendix D.1'
  'L.AE'     = 'Appendix D.2'
  'S.DEMO.A' = 'Appendix C.1'
  'S.DEMO.E' = 'Appendix C.2'
  'S.AE.A'   = 'Appendix C.22'
  'S.AE.R'   = 'Appendix C.23'
   ;
run;
```

By defining a macro variable in each program which designates the program (&PGM), a cross reference can be made using the $TABLE format to determine the appendix number. To help with consistency, using the same code in the PGM variable as the program name provides a means of locating the program from the hard copy if the need arises. The PGM variable would be set prior to the DATA _NULL_ step of your program using the following syntax:

```
%let PGM = L.DEMO;
```

The last %IF portion of the HDG macro shows how to use the PUT function with the $TABLE format to get a title containing the appropriate appendix number. This can then be displayed using a PUT statement.

PATIENT IDENTIFYING INFORMATION (TOPLIST MACRO)

The patient identifying information is usually displayed consistently across all listings. The sort order of these is likely to be the same as well (although each table may have additional sorting variables). For example if the top of each listing is to display the investigator name, patient's treatment group, patient number, patient initials, and evaluabilty code, a macro can be defined to do this in the same manner for each listing. Below is an example of the TOPLIST macro which assumes the data coming into the DATA _NULL_ is sorted by at least investigator name, treatment group, and patient number:

```
%macro TOPLIST;
  if first.PAT or TOP then do;
    if first.TRTMNT or TOP then do;
      if first.INV or TOP then do;
        if TOP or (_n_ eq 1) then
          put @1 'Investigator: '
                 INV;
        else
          put / @1 'Investigator: '
                   INV;
      end;
      if not(TOP or (first.INV)) then put /;
      put @1 'Treatment Group: '
             TRTMNT;
    end;

    if first.PAT then PTCOUNT + 1;

    put %PUTFLD(PAT)
        %PUTFLD(INIT)
        %PUTFLD(EVALCODE) @;
 end;
%mend;
```

This TOPLIST macro is then called by each data listing program to insure that the patient identifying information is displayed in the same manner across listings. The use of a nested IF structure is efficient and easy to follow. Due to the sort sequence, there can never exits a FIRST.INV without a FIRST.TRTMNT. The TOP variable is defined in the Final DATA _NULL_ to be a variable which is set each time a new page is begun. This way each new page will have the patient identifying information regardless of whether it is the first observation for that patient. Below is a DATA _NULL_ showing how each program would simply call this TOPLIST macro:

```
data _null_;
  set rpt end=EOF;
    by INV TRTMNT PAT {other vars};
    ...
    %TOPLIST;
    ...
run;
```

This eliminates the chances of different programmers coding this section inconsistently. The PUTFLD macro used in this TOPLIST macro will be described in more detail below.

DISPLAYING EACH DATA FIELD (COL, FMT, AND PUTFLD MACROS)

Determine, for each display, the column where each data item is to begin. By defining a COL macro to define column numbers for each data item, both the column titles as well as for the actual data items can reference the same macro. Below is an example of a COL macro:

```
%macro COL(VAR);
  %let &VAR = %upcase(&VAR);
  %if       &VAR eq PAT      %then 3;
  %else %if &VAR eq INIT     %then 12;
  %else %if &VAR eq EVALCODE %then 21;
  %else %if &VAR eq SEX      %then 35;
  %else %if &VAR eq AGE      %then 42;
  %else %if &VAR eq RACE     %then 50;
  %else %if &VAR eq HEIGHT   %then 65;
  %else %if &VAR eq WEIGHT   %then 75;
  %else put "ERROR-NO DECODE FOR &VAR";
%mend;
```

This macro is driven by the variable name. Given a particular variable name, this macro will return a number which indicates the column in which the data is to begin printing. This data, as mentioned above, can be the actual data, or can also be a trigger for the column headings as well. For example, the following code:

```
put @%COL(PAT)      'Patient'
    @%COL(INIT)     'Patient'
    @%COL(EVALCODE) 'Evaluability'
  / @%COL(PAT)      'Number'
    @%COL(INIT)     'Initials'
    @%COL(EVALCODE) '    Code';
```

This would cause the titles to be printed in the columns specified by the COL macro.

There are many situations when the data coming into the data null is not formatted exactly as desired. For example, SEX may be denoted by an 'F' or 'M'. On the display, this would look nicer if it read 'Female' and 'Male'. Hence using a macro like the FMT macro below, formats can be assigned to each variable as necessary:

```
%macro FMT(VAR);
  %let &VAR = %upcase(&VAR);

  %if      &VAR eq EVALCODE %then EVAL.;
  %else %if &VAR eq SEX     %then SEX.;
  %else %if &VAR eq AGE     %then 3.0;
  %else %if &VAR eq RACE    %then RACE.;
  %else %if &VAR eq HEIGHT %then 5.1;
  %else %if &VAR eq WEIGHT %then 5.1;
%mend;
```

There need not be an entry in the FMT macro for every field. This would mean that the PUT statement would not have a format associated with it, which is not an error. Now that both the column and the format for each data item have been defined, a final macro is defined to pull these two together. This macro is called PUTFLD, since it is part of a PUT statement and is displaying a given data FIELD:

```
%macro PUTFLD(NAME,OFFSET=);
  %if "&OFFSET" eq ""
     %then @%COL(&NAME)
                 &NAME
            %FMT(&NAME);
     %else @%COL(&NAME) + &OFFSET
                 &NAME
            %FMT(&NAME);
%mend;
```

This PUTFLD macro calls the COL macro to determine the column to begin "putting" the data and calls the FMT macro to determine the format to make the display of the data items come together. The OFFSET parameter is helpful in that it tells the PUTFLD macro to move your data items over in the display for any given reason. If the PUTFLD macro is called with an OFFSET=2, the data item will be displayed two columns to the right of the column indicated by the result of the COL macro for that variable. The most common reason would be to align data nicely under a longer column title. Below is an example of the DATA _NULL_ step which utilizes the PUTFLD macro:

```
data _null_;
  set rpt end=EOF;
    by INV TRTMNT PAT {other vars};
    ...
    %TOPLIST;
    put %PUTFLD(SEX)
        %PUTFLD(AGE,OFFSET=1)
        %PUTFLD(RACE)
        %PUTFLD(HEIGHT)
        %PUTFLD(WEIGHT);
    ...
run;
```

In the above DATA _NULL_ step the TOPLIST macro and the PUTFLD macro have been shown. The PUTFLD macro is called from within a PUT statement. This allows for the displaying of multiple fields within one PUT statement. This is especially handy when the fields are related.

THE FOOTING (FOOT MACRO)

The footing of each listing may be the same across the board, or there may be just parts of it that are similar. Below is an example of a footing on listings where there is a solid line delimiting the actual data, then a legend type footer indicating what an "1" and an "2" stand for in the body of the listing.

```
%macro FOOT(FID,
            FNCT=FNCT);
  do while (ll ge (&fnct+1));
    put;
  end;

  put &LS*'_' /;
  put @1 '1=Evaluable, 2=Not Evaluable';
%mend;
```

The first item to note in the macro is the PS macro variable. This is a global variable which was set prior to the DATA _NULL_ step which defined the number of lines on the page, or pagesize. The FNCT macro variable is the variable which contains the number of lines necessary to display all the footnote lines. This footer assumes that EVERY listing will have both the solid line and the legend text for "1" and "2". Additional footer lines can be coded right into the individual programs, or a select statement based on the FID macro variable can be used. For example:

```
%macro FOOT(FID,
            FNCT=FNCT);
  do while (ll ge (&fnct+1));
    put;
  end;

  put &LS*'_' /;

  select("&FID");
    when("L.AE")
      put @1 'Patients not included on'
             ' this listing have no'
             ' Adverse Events';
    otherwise
      put @1 '1=Evaluable, '
             '2=Not Evaluable';
  END;
%mend;
```

This example shows that for the L.AE listing, the "1" and "2" legend text should not be displayed. Instead, a description of why all the patients are not included in the display is printed. Keeping the more generic footnotes in this FOOT macro make it much easier to change them if there happens to be a typographical error, or if there is need for a change in a footnote that appears in multiple displays.

THE END OF THE DISPLAY (EOF MACRO)

When the last record in the dataset has been read, some final processing needs to be done. The program has been keeping a count of the number of patients being included in the listing (via the TOPLIST macro), hence, displaying this total number of patients can be done. Since there is no more data to process, the program can display text indicating this is the last appendix page. Below is a macro which will take care of these two final issues:

```
%macro EOF;
  if EOF then do;
    put #1 @%eval(&LS-39) 'LAST APPENDIX';
    link FOOT;
    put _PAGE_;
    put //@10 'Total Patients: '
              PTCOUNT;
  end;
%mend;
```

First, there is a test to see if the end of the file has been reached. Note that in order to determine this, there must be and END=EOF option in the SET statement of the DATA _NULL_ step. If the end of the file has been reached, the footer is displayed, and on the first line of the current display page, the words LAST APPENDIX are added to the PAGE number line. Then a new page is begun and the only thing displayed on this page is the patient count. This patient number count be helpful to be sure that all listings contain information for the same number of patients. Any other data counts that are important to the data listing can be maintained in the code and displayed in this manner.

THE DATA _NULL_

To pull all of these macros together, one DATA _NULL_ can be used to show how these macros all interact. Below is the DATA _NULL_:

```
data _null_;
  set rpt end=EOF;
    by INV TRTMNT PAT {other vars};

    file print header=HDG notitles LL=ll
               LS=&ls PS=&ps N=ps;

    if (LL lt (FNCT+1)) or
       (first.TRTMNT and not(TOP) and
        (LL lt (LINES+FNCT+1))) then do;
      link FOOT;
      put _PAGE_;
    end;

    %TOPLIST;

    put %PUTFLD(SEX)
        %PUTFLD(AGE,OFFSET=1)
        %PUTFLD(RACE)
        %PUTFLD(HEIGHT)
        %PUTFLD(WEIGHT);
    %EOF;
return;

HDG:
  TOP = 1;
  PNUM + 1; /* increment page number */
  FNCT = 3; /* number of footnote LINEs */
  retain FNCT;

  %HDG(&PGM);

  put %CTRPUT("&TTL");

  put @1 &LS*'_' /
    / @%COL(PAT)       'Patient'
      @%COL(INIT)      'Patient'
      @%COL(EVALCODE)  'Evaluability'
      @%COL(AGE)       'Age'
      @%COL(HEIGHT)    'Height'
      @%COL(WEIGHT)    'Weight'
    / @%COL(PAT)       'Number'
      @%COL(INIT)      'Initials'
      @%COL(EVALCODE)  '    Code'
      @%COL(SEX)       'Sex'
      @%COL(AGE)       '(yrs)'
      @%COL(RACE)      'Race'
      @%COL(HEIGHT)    ' (cm)'
      @%COL(WEIGHT)    ' (kg)'
    / @1 &LS*'_' /;
return;


FOOT:
  %FOOT(&PGM);
  put @1 'Subject 0101 Randomized Twice';
return;
run;
```

This DATA _NULL_ incorporates all the macros which have been presented within this paper. The FILE statement is partial but includes the items necessary for the macros to run. The PS and LS macro variables referenced here are global variables indicating page size and line size. In the FILE statement, the HEADER=HDG refers to the lines of code which are executed at the top of each page. These are found at the HDG: label below the RETURN statement of the DATA _NULL_. These are not to be confused with the HDG macro which is called from this section of code. The code in the FOOT section is called at the end of each page (code not included here), and at the end of the file as well. At the end of this paper, there is an example of the output produced by this DATA _NULL_ step.

MODIFYING THE DISPLAY

For one reason or another, there are always changes to be made to the listings. By using these macros, changes to the displays can be made quickly and easily. For example, if the RACE field (decoded) is taking up too many columns for the display to look nice, a simple change to the column numbers in the COL macro will adjust for this change. A simple re-run of the program will automatically make the changes to the titles as well as to the data itself. Another example would be the addition of a data field. This would be done using the following simple steps:

- add an entry to the COL macro for the new field.
- modify COL numbers to separate out all the fields with this new addition.
- add an entry to the FMT macro for the new field.
- in the DATA _NULL_, add a PUTFLD call for the new field.
- add the necessary title information to the heading of the DATA _NULL_ for the new field.

Again, a re-run will then display the new field, with all other fields shifted properly.

SUMMARY

The PGM macro variable, the COL macro and the FMT macro are specific to each program and must reside within each program. All other macros can be placed in an included piece of code, or they may reside in a SAS MACRO library. These macros are very helpful in making the displays across an entire project (and project team) consistent. They make changing a display very easy as well as quick. These macros all work together to provide a fine tool in data table generation. They allow maximum flexibility while adhering to standard programming techniques.

AUTHOR INFORMATION

The SAS source code contained within this paper was developed by the following programmers at IBAH, Inc., formerly Bio-Pharm Clinical Services, Inc.: Mike Mercurio, Larry Specht, and Daphne Ewing.

Daphne Ewing
Synteract, Inc.
714 N. Bethlehem Pike, Suite 300
Ambler, PA 19002
(215) 283-9470
(215) 283-9366 (fax)
dewing@synteract.com
www.synteract.com

```
PhunnyPharm, Inc.                                                                                                   PAGE 2
Protocol No. ABC-123
Clinical/Statistical Report                                                                              L.DEMO 13JAN02 06:043
                                                                Appendix D.1

                                                       Listing of Patient Demographics
______________________________________________________________________________________________________________________________________________

  Patient    Patient   Evaluability               Age                 Height   Weight
  Number    Initials       Code         Sex      (yrs)      Race       (cm)      (kg)
______________________________________________________________________________________________________________________________________________

Investigator: 0013
Treatment Group: Better Drug
 0101       CLE        Non-Evaluable  Female      28     Caucasian   68.0      145.0
 0103       ATM        Evaluable      Female      34     Black       63.5      197.0
 0105       TDM        Evaluable      Male        26     Caucasian   74.0      181.0
 0107       CAH        Evaluable      Female      42     Black       69.5      184.0
 0301       LRH        Evaluable      Female      33     Caucasian   66.0      150.0
 0303       CCP        Evaluable      Male        21     Black       67.0      172.0
 0305       BAK        Evaluable      Female      31     Caucasian   70.0      197.0
 0307       NLH        Evaluable      Female      20     Black       61.0      128.0
 0309       KAD        Evaluable      Male        29     Caucasian   72.5      180.0
 0311       PCA        Evaluable      Female      36     Caucasian   60.0      144.0
 0313       CMB        Evaluable      Female      44     Caucasian   67.0      168.0
 0315       LJZ        Evaluable      Male        35     Caucasian   71.5      150.0
 9003       DJP        Non-Evaluable  Male        48     Caucasian   69.0      199.0
 9023       GMJ        Non-Evaluable  Female      31     Caucasian   70.0      138.0

Investigator: 0014
Treatment Group: Good Drug
 0118       DCB        Evaluable      Female    32       Black        60.0      165.0
 0120       RNM        Evaluable      Male      27       Black        63.0      155.0
 0122       AJG        Evaluable      Female    26       Black        68.0      134.0
 0124       TAW        Evaluable      Female    29       Black        68.5      156.0
 0358       SLR        Evaluable      Female    22       Black        63.0      130.0
 9002       TDG        Non-Evaluable  Female    23       Black        64.0      160.0
 9008       MMU        Non-Evaluable  Female    29       Asian        61.0      102.0



______________________________________________________________________________________________________________________________________________

1=Evaluable, 2=Not Evaluable
```