Macros: Data Listings with Power
Daphne Ewing, Auxilium Pharmaceuticals, Inc., Malvern, 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 regulatory agencies, 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. The same philosophy can be used for data summaries using the DATA_NULL_reporting method.

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 subject identifying information and the order in which it is displayed. The portions of the displays that change are usually the different data items specific to each display (i.e. Demography fields versus Adverse Event fields, etc.). 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, statistical, ISS, etc.), 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 across all listings. Below is an example of a heading macro:

```sas
%macro hdg(HID, PNUM, TBLNO, TITLE);
put @1 'Phunny Pharm, 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 "&TBLNO" ne ""
%then &TBLNO = put("&HID",$TABLE.)%str(;;);
put;
%ctrput(&TBLNO);
%end;
%if "&TITLE" ne ""
%then &TITLE = put("&HID",$TBLTTL.)%str(;;);
%ctrput(&TITLE);
%end;
%mend hdg;
```

This HDG macro is comprised of the PUT statements necessary to display the heading information for all tables in a consistent manner. In the example above, the name of the company, left justified, and page number, right justified, appear on the first line. There is reference here to the LS macro variable, &LS, which is a global macro variable which is set prior to the DATA_NULL_step that defines the number of spaces to a line, or linesize. The protocol appears on the second line, also left justified, and on the third line is the type of the report, left justified, and the program name that created this report (&PGM, defined at the top of each program to uniquely identify the program), the system date and the system time, right justified. The last portion of this macro shows how to display the appendix number and the table title. This code uses the PUT function with the $TABLE and $TBLTTL format to get titles containing the appropriate appendix number and table title. These can then be displayed using a PUT statement (the call to CTRPUT macro uses the put statement to center justify the values decoded from the PUT function and will be displayed in more detail below). This HDG macro should be defined at the start of a project so that all output for a project can be consistent.
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. Although the Table titles do not change too often, this same philosophy to appendix numbers can be applied to table titles. Below is an example of $TABLE and TTBLTTL formats that will be incorporated into the table creation process:

```
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';
  value $TBLTTL
    'L.DEMO'  = 'Listing of Subject Demographics'
    'L.AE'    = 'Listing of All Adverse Events'
    'S.DEMO.A' = 'Summary of Demographics - All Subjects'
    'S.DEMO.E' = 'Summary of Demographics - Evaluable Subjects'
    'S.AE.A'   = 'Summary of All Adverse Events'
    'S.AE.R'   = 'Summary of Treatment Related Adverse Events';
run;
```

By defining a macro variable in each program which designates the program (&PGM), a cross reference can be made using the $TABLE or $TBLTTL format to determine the appendix number or table title. 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;
```

This macro variable definition is best defined at the top of the program near other setup items in the program.

**DISPLAYING CENTERED TITLES(CTRPUT MACRO)**

With many titles, it is more pleasing to the eye to center justify the text. With TITLE statements in SAS, this is easy to do, as it is the default format for the TITLE statement. With DATA _NULL_ reporting, this is not the case. The CTRPUT macro below allows the text passed in as a parameter to be displayed in a center justified manner. It is important that the LS macro variable be defined prior to using this macro.

```
%macro ctrput(TXT);
  %if "&TXT" ne "" %then %do;
    put @((&LS/2) - int(lengthn(&TXT)/2)) &TXT;
  %end;
%mend ctrput;
```

This macro simply does the mathematical calculation in order to start the output of the text string at an appropriate location so as to center justify the string.

**SUBJECT IDENTIFYING INFORMATION (TOPLIST MACRO)**

The subject 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, subject's treatment group, subject number, subject initials, and evaluability 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 subject number:

```
%macro toplist(PTCNT=SUBJCNT);
  if first.SCRNO or TOP then do;
    if first.TRTMNT or TOP then do;
      if first.CENTNO or TOP then do;
        if TOP or (_n_ eq 1) then
          put @1 'Investigator: ' invname;
        else
          put // @1 'Investigator: ' invname;
      end;
    end;
    put @3 'Treatment Group: ' TRTMNT;
  end;
  ** Count number of subjects for QC purposes **;
  if first.SCRNO then &PTCNT + 1;
%mend toplist;
```

And Now, Presenting...
** Display "header" type variables **;
put %put fld(SCRNO)
%putfld(PATINIT)
%put fld(EVALCODE) @;
end;
%mend toplist;

This TOPLIST macro is then called by each data listing program to insure that the subject 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 be 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 subject identifying information regardless of whether it is the first observation for that subject. Below is a DATA _NULL_ showing how each program would simply call this TOPLIST macro:

data _null_;  
set db.rpt end=EOF;  
by CENTNO TRTMNT SCRNO {other vars};  
...  
%TOPLIST;  
...  
run;

The use of this TOPLIST macro 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:

```plaintext
%macro col(VAR);
%let &VAR = %upcase(&VAR);
%if &VAR eq SCRNO %then 7;
%else %if &VAR eq PATINIT %then 16;
%else %if &VAR eq EVALCODE %then 27;
%else %if &VAR eq SEX %then 45;
%else %if &VAR eq AGE %then 55;
%else %if &VAR eq RACE %then 64;
%else %if &VAR eq HEIGHT %then 84;
%else %if &VAR eq WEIGHT %then 96;
%else put // "ERR" "OR: NO DECODE FOR &VAR" //;
%mend col;
```

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:

```plaintext
put @1 &LS*'_' /  
@&COL(SCRNO) "Subject'  
@&COL(PATINIT) 'Subject'  
@&COL(EVALCODE) 'Evaluability'  
...  
/ @&COL(SCRNO) 'Number'  
@&COL(PATINIT) 'Initials'  
@&COL(EVALCODE) + 3 'Code [a]'  
...  
/ @1 &LS*'_' /;
```

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 '1' or '2'. On the display, this would look nicer if it read 'Male' and 'Female'. Hence using a macro like the FMT macro below, formats can be assigned to each variable as necessary:

```plaintext
%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;
%mend fmt;
```
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 putfld;
```

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 db.rpt end=EOF;  
  by CENTNO TRTMNT SCRNO;  
  ...  
  %TOPLIST;  
  put %putfld(SEX)  
       %putfld(AGE,OFFSET=1)  
       %putfld(RACE)  
       %putfld(HEIGHT)  
       %putfld(WEIGHT)  
;  
run;
```

In the above DATA _NULL_ step the TOPLIST macro call and the PUTFLD macro calls 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 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 the values of "1" and "2" stand for in the body of the listing.

```
%macro foot(FID,  
FNCT=FNCT);  
do while (LNSLEFT gt &FNCT);  
  put;  
  end;  
  put &LS*'_';  
  put &1 ' [a] 1 = Evaluable, 2 = Not Evaluable';  
%mend foot;
```

The FNCT macro variable is the variable which contains the number of lines necessary to display all the footnote lines for this display. 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 (LNSLEFT gt &FNCT);  
  put;  
  end;  
  put &LS*'_';
```
select("&FID");
when("L.AE")
  put @1 'Subjects not included on this listing have no Adverse Events';
otherwise;
end;
put @1 '[a] 1 = Evaluable, 2 = Not Evaluable';
%mend foot;

This example shows that for the program with &PGM of 'L.AE', the "1" and "2" legend text should be displayed after the information about why subjects may not be on the listing. 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 may need to be done. The program has setup to keep a count of the number of subjects being included in the listing (via the TOPLIST macro), hence, displaying this total number of subjects 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(PTCNT=SUBJCNT);
  if EOF then do;
    put #1 @%eval(&LS - 39) 'LAST APPENDIX';
    link FOOT;
    put _PAGE_;
    put //@10 'Total Subjects: ' &PTCNT;
    link FOOT;
    end;
%mend eof;

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 an 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 subject count. This subject number count may be helpful to be sure that all listings contain information for the same number of subjects. 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 db.rpt end=EOF;
by CENTNO TRTMNT SCRNO;
file print header=HDG notitles LL=LNSLEFT LS=&ls PS=&ps N=ps;
  retain FNCT 4;  ** Set to total number of rows used for footnotes **;
  if (LNSLEFT lt (FNCT + 1)) or
    (first.TRTMNT and not(TOP) and
      (LNSLEFT lt (LINES + FNCT + 1))) then do;
      link FOOT;
      put _PAGE_;
      put //@10 'Total Subjects: ' &PTCNT;
      link FOOT;
    end;
%mend TOPLIST;
  put %putfld(SEX)
    %putfld(AGE,OFFSET=1)
    %putfld(RACE)
    %putfld(HEIGHT)
    %putfld(HEIGHT)
    ;
%mend eof;
return;
HDG:
TOP = 1; /* flag indicating top of page executed */
PNUM + 1; /* increment page number */

%HDG(&PGM);

put @1 &LS*'_' /
/ @%COL(SCRNO) 'Subject'
@%COL(PATINIT) 'Subject'
@%COL(EVALCODE) 'Evaluability'
@%COL(Age) + 1 'Age'
@%COL(HEIGHT) 'Height'
@%COL(WEIGHT) 'Weight'
/ @%COL(SCRNO) 'Number'
@%COL(PATINIT) 'Initials'
@%COL(EVALCODE) + 3 'Code [a]'
@%COL(SEX) 'Sex'
@%COL(Age) + 1 '(yrs)'
@%COL(RACE) 'Race'
@%COL(HEIGHT) + 1 '(cm)'
@%COL(WEIGHT) + 1 '(kg)'
/ @1 &LS*'_' /;
return;

FOOT:
%FOOT(&PGM);
put @1 'Subject 1002 Randomized Twice';
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 (HDG area) 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.

CONCLUSION
Given the option of a horizontal versus a vertical data structure to be used during the analysis phase of clinical trials, there are many reasons to choose the vertical structure. If your company has not gone to "standards" at this point, it might be worth looking into a normalized data structure as your standard. This design provides for flexibility and makes standard programs more robust. The earlier on in the data process where the data can be normalized, the analysis process benefits are greater.
REFERENCES

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. © indicates USA registration.

Other brand and product names are trademarks of their respective companies.

ACKNOWLEDGMENTS
The SAS source code contained within this paper was developed by several programmers at IBAH, Inc. (now known as OmniCare Clinical Research) and further refined by the author in subsequent years.

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

Daphne Ewing, Auxilium Pharmaceuticals, Inc.
Voice: (215) 740-6503
FAX: (484) 321-2269
Email: DEwing@auxilium.com
## Appendix D.1
### Listing of Subject Demographics

<table>
<thead>
<tr>
<th>Subject Number</th>
<th>Initials</th>
<th>Evaluability</th>
<th>Age (yrs)</th>
<th>Sex</th>
<th>Race</th>
<th>Height (cm)</th>
<th>Weight (kg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1101</td>
<td>S-C</td>
<td>Evaluable</td>
<td>64</td>
<td>Male</td>
<td>Hispanic</td>
<td>175.3</td>
<td>113.4</td>
</tr>
<tr>
<td>1103</td>
<td>GTS</td>
<td>Not Evaluable</td>
<td>75</td>
<td>Male</td>
<td>Caucasian</td>
<td>172.7</td>
<td>87.3</td>
</tr>
<tr>
<td>1105</td>
<td>DYG</td>
<td>Not Evaluable</td>
<td>68</td>
<td>Male</td>
<td>Caucasian</td>
<td>170.2</td>
<td>94.9</td>
</tr>
</tbody>
</table>

**Investigator: Dr. Jones (1039)**
- Treatment Group: Active

<table>
<thead>
<tr>
<th>Subject Number</th>
<th>Initials</th>
<th>Evaluability</th>
<th>Age (yrs)</th>
<th>Sex</th>
<th>Race</th>
<th>Height (cm)</th>
<th>Weight (kg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2704</td>
<td>E2M</td>
<td>Not Evaluable</td>
<td>54</td>
<td>Male</td>
<td>Caucasian</td>
<td>160.0</td>
<td>73.0</td>
</tr>
<tr>
<td>2705</td>
<td>JPH</td>
<td>Not Evaluable</td>
<td>42</td>
<td>Male</td>
<td>Caucasian</td>
<td>185.4</td>
<td>109.4</td>
</tr>
</tbody>
</table>

**Investigator: Dr. Lyons (1093)**
- Treatment Group: Comparator

**Investigator: Dr. Smith (1095)**
- Treatment Group: Active

<table>
<thead>
<tr>
<th>Subject Number</th>
<th>Initials</th>
<th>Evaluability</th>
<th>Age (yrs)</th>
<th>Sex</th>
<th>Race</th>
<th>Height (cm)</th>
<th>Weight (kg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1002</td>
<td>ARW</td>
<td>Not Evaluable</td>
<td>51</td>
<td>Male</td>
<td>Caucasian</td>
<td>175.3</td>
<td>96.8</td>
</tr>
<tr>
<td>1009</td>
<td>J-A</td>
<td>Evaluable</td>
<td>46</td>
<td>Male</td>
<td>Black</td>
<td>182.9</td>
<td>87.9</td>
</tr>
<tr>
<td>1014</td>
<td>MSC</td>
<td>Not Evaluable</td>
<td>54</td>
<td>Male</td>
<td>Caucasian</td>
<td>186.0</td>
<td>78.5</td>
</tr>
<tr>
<td>1017</td>
<td>JMH</td>
<td>Not Evaluable</td>
<td>55</td>
<td>Male</td>
<td>Caucasian</td>
<td>172.7</td>
<td>73.4</td>
</tr>
<tr>
<td>1025</td>
<td>GJK</td>
<td>Not Evaluable</td>
<td>60</td>
<td>Male</td>
<td>Caucasian</td>
<td>182.9</td>
<td>119.3</td>
</tr>
<tr>
<td>1026</td>
<td>J-B</td>
<td>Not Evaluable</td>
<td>53</td>
<td>Male</td>
<td>Hispanic</td>
<td>180.3</td>
<td>109.1</td>
</tr>
<tr>
<td>1030</td>
<td>WCM</td>
<td>Not Evaluable</td>
<td>71</td>
<td>Male</td>
<td>Caucasian</td>
<td>175.3</td>
<td>102.4</td>
</tr>
<tr>
<td>1037</td>
<td>MCL</td>
<td>Not Evaluable</td>
<td>57</td>
<td>Female</td>
<td>Hispanic</td>
<td>170.2</td>
<td>79.4</td>
</tr>
<tr>
<td>1038</td>
<td>AST</td>
<td>Not Evaluable</td>
<td>57</td>
<td>Male</td>
<td>Hispanic</td>
<td>177.8</td>
<td>106.9</td>
</tr>
<tr>
<td>1041</td>
<td>GMW</td>
<td>Not Evaluable</td>
<td>69</td>
<td>Male</td>
<td>Hispanic</td>
<td>172.7</td>
<td>79.8</td>
</tr>
</tbody>
</table>

[a] 1 = Evaluable, 2 = Not Evaluable

Subject 1002 Randomized Twice