Paper 210-2010

SAS® Macro to Automate Data Listings for New Drug Applications

Hany Aboutaleb, Biogen Idec, Cambridge, MA

Abstract:

Using the SAS® Macro language to generate data listing programs eliminates the need to create these programs from scratch. In most cases the programs are simple – consisting primarily of proc report statements. Using the SAS® Macro language to generate these statements automatically can significantly reduce the amount of time required to write these programs. Once the listing programs are generated, it can be edited, and modified.

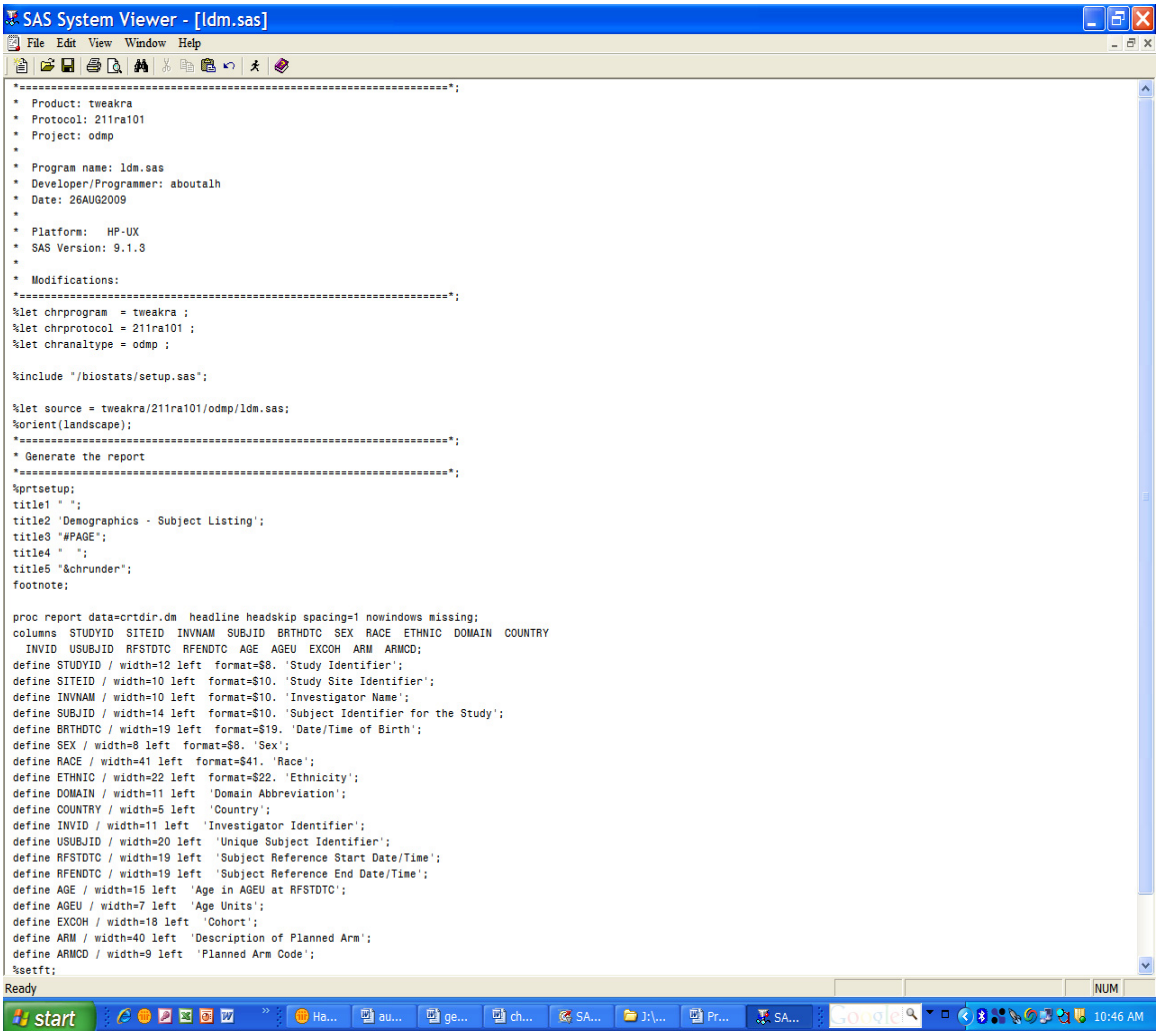
Macro %Gen-list Logic:

The %Gen-list macro requires the library reference to your data (i.e., libin: the library name where the SAS dataset resides), and name of the dataset (i.e., dsn). If you would like to produce one data list you should choose dsn=DM or if you would like to produce all the data listings choose _all_. The macro default value for the dsn parameter _all_ is used to generate the whole library listing. Another option to store your generated programs is outdir (i.e., the output directory to store the generated program). The macro uses the contents procedure to identify the name of each dataset and each variable and attribute. These will be used to create a SAS program consisting of proc report statements using the information that was gathered from the contents procedure. The output generated program name for the data listing will consist of lowercase l followed by the data set name and with extension sas (e.g., ldm.sas).

A SAS listing program consisting of the following statements were generated from the results of a Proc Contents. (See Figure 1)

- A header details the protocol, program name, directory path, program creator, creation date, platform, SAS version, and any modifications that have been made to the code.
- A title statement containing the SAS dataset label name.
- A footnote statement containing the source and date the program ran.
- A proc report statement with appropriate parameters.
- A column statement followed by a list of variables in order by variable number.
- A define statement for each variable with the following parameters:
  - Width derived from length and format Length variables
  - Column heading derived from label
  - Format (if Variable is formatted)

Figure 1 – SAS generated listing program



Once the listing program is generated, it can be edited to:

- Modify title/footnote statements.
- Eliminate unwanted variables.
  - If code and decode variables are present usually only the decode variable is displayed.
- Define order.
- Change width.
  - If the variable is numeric and unformatted, the width defaults to 8.
  - If the variable length is very large, reduce width and add flow parameter.
- Change column heading to be more descriptive.

Macro Input Parameters:

libin: Name of Library where SAS Dataset resides
dsn: Name of SAS Dataset (default = _all_)
outdir: Output directory to store the generated program
version: Version control of the macro for future use, in case of new release of the macro (default = 1)
debug: To debug the macro YES/NO (default = NO)

Sample Call:

%gen_list (libin=crtdir, dsn=dm,outdir=/biostats/cdisc/qc)

Conclusion:

The Macro %gen_list is very easy to use, modify or update and take almost no time to run. Once the listing program is generated it can be edited to modify the title, footnote, define order, width, column heading, and to eliminate unwanted variables.

TRADEMARKS:

REFERENCES:

[1] *SAS® 9.2* Macro Language: Reference. Cary, NC: SAS Institute Inc
[2] *SAS® 9.2* Language Reference: Concepts. Cary, NC: SAS Institute Inc
[3] *SAS® 9.2 Language Reference: Dictionary*. Cary, NC: SAS Institute Inc.

Contact Information

Your comments and questions are valued and encouraged.

Contact the author at:

Hany Aboutaleb
Biogen Idec.
14 Cambridge Center
Cambridge MA 02142
Tel. #: (617) 914-7125
Fax: (617) 679-3280
Internet: hany.aboutaleb@biogenidec.com

Macro Code:

```
%*-------------------------------------------------------------------
 Program ID:          gen list.sas
 Source pgm:          None

 Description:         Generate needed SAS statements or programs to
                          generate a listing or listings
 Created:             August 25, 2009

 Developer:           Hany Aboutaleb (BiogenIdec, 47125)
 Programmer:          Hany Aboutaleb (BiogenIdec, 47125)
 Contact persons:     Macro Librian (Biogenidec, 47125)

Input parameters:     libin:   name of Library where SAS Dataset resides
                       dsn:     name of SAS Dataset
                               (default= all )
                       outdir:  output directory to store the generated program
                       Version: Version control to the macro for future use in case of new
                                release to the macro
                               (default: 1)
                      debug:    to debug the macro (yes/no)
                                                  (default: NO)
 Sample Call:         %gen list (libin=crtdir,
                                 dsn=dm,outdir=/biostats/tweakra/211ra101/cdisc/qc)

 Note:                Please contact the Macro Liberian for any suggestion or comments

 Modification log:
 ----------------
 Date     Reason                                                  Person
 ----     ------                                                  ------

-------------------------------------------------------------------------------------------;

%macro gen list (libin=,
                    dsn= all_,
                 debug=N,
               VERSION=1,
                outdir=);

%put;
%put<<<=============================================================================================>>>;
%put <> Biogen IDEC SYSTEM MACRO gen list: Generate a SAS listing program automatically;
%put<<<=============================================================================================>>>;
%put;

%if %upcase(&debug)=YES %Then
     %do;
        options symbolgen mlogic mprint;
              %put _user_;
     %end;

%local parmerr er ror sep;
     %let parmerr=0;
     %let er  = ER ;
     %let ror = ROR ;
  %*-----------------------------------------------------------------------
  Determine the OS specific directory command
 ------------------------------------------------------------------------;
   %if %scan(&sysscp,1) = HP %then
     %do;
      %let sep=/;
     %end;
       %else %do;
         %let sep=\;
       %end;
%*-----------------------------------------------------------------------
  Validate parameters
-------------------------------------------------------------------------;
%let parmerr=0;

** Break if parameter libin empty ;
  %if ( %sysfunc(libref( &libin )) ne 0 ) %then
    %do;
       %put %sysfunc(sysmsg());
       %put &er&ror: Input library %str(&libin) is not correctly defined.;
       %let parmerr=1;
   %end;
```

```
** Break if parameter dsn empty ;
  %if %sysevalf(%superq(dsn)=,boolean) %then
    %do;
       %put %sysfunc(sysmsg());
       %put &er&ror: Input Data set %str(&dsn) is not correctly defined.;
       %let parmerr=1;
   %end;
%*-------------------------------------------------------------------
  Quit the macro program if you have any wrong input parameters
-------------------------------------------------------------------;
%if (&parmerr) %then %return;
%*-------------------------------------------------------------------------;
%if &version=1 %then
    %do;
%if ^%sysevalf(%superq(libin)=,boolean) and ^%sysevalf(%superq(dsn)=,boolean) %then
  %do;
%*-------------------------------------------------------------------------
  Using Proc Contents - Generate a file containing a List of SAS Dataset Names
-------------------------------------------------------------------------;
  proc contents data=&libin..&dsn out=content(keep=memname) noprint;
  proc sort data=content out=conts nodupkey;
       by memname;
  run;
%*-------------------------------------------------------------------
  Using the list of SAS Dataset Names to Generate a list macro variables
-------------------------------------------------------------------;
  data  null ;
     set conts end=last;
      by memname;
     call symput('ds'||left( n ),trim(lowcase(memname)));
     if last then call symput('tot',left(_n_));
  run;
 %end;
%do i=1 %to &tot;
%*-------------------------------------------------------------------
 Generate a List of Variables and Attributes in the SAS Dataset
-------------------------------------------------------------------;
proc contents data=&libin..&&ds&i out=dsn_list noprint;
 run;

proc sort data=dsn_list;
  by varnum;
run;
%*-------------------------------------------------------------------
logic to adjust the variables width for the report define statement
-------------------------------------------------------------------;
data dsn list;
       set dsn list;
        by varnum;
    varwid=length(trim(left(label)));
    if length>8 then nlength=round(((length+varwid)/3),1);
      else nlength=round(((length+varwid)/2),1);
    if length > nlength then vlength=length;
      else vlength=nlength;
run;
%*-------------------------------------------------------------------
 create the macro variables to hold the column values
-------------------------------------------------------------------;
proc sql noprint;
        select name into: colum separated by ' '
        from dsn_list ;
quit;

%if %upcase(&debug)=YES %Then
     %do;
               %put _user_;
     %end;
%*-------------------------------------------------------------------
 Generate a SAS File containing SAS Statements and Sequence Numbers
-------------------------------------------------------------------;
data list dsn (keep=rpt line stmnt_num);
   set dsn list end = eof;
   by memname;

  length stmnt num 8.;   /* Defines the order of the SAS statements */
  length rpt_line $200.; /* Contains the SAS Statement */
```

```
%*--------------------------------------
  When reading the first record
  Create stand alone program header
 --------------------------------------;
  if  n_ eq 1 then
   do;
    stmnt num = 0.001;
    rpt line='*================================================================================*;';
     output;
         stmnt num = 0.002;
     rpt line="*  Product: &chrprogram ";
     output;
         stmnt num = 0.003;
     rpt line="*  Protocol: &chrprotocol ";
     output;
         stmnt num = 0.004;
     rpt line="*  Project: &chranaltype ";
     output;
         stmnt num = 0.005;
     rpt line='* ';
     output;
         stmnt num = 0.006;
     rpt line="*  Program name: l&&ds&i...sas";
     output;
         stmnt num = 0.007;
     rpt line="*  Developer/Programmer: &sysuserid ";
     output;
         stmnt num = 0.008;
     date1 = put(today(),date9.);
     rpt line='*  Date: ' || date1;
     output;
     stmnt num = 0.009;
     rpt line='* ';
     output;
         stmnt num = 0.010;
     rpt line='*  Platform:   HP-UX';
     output;
         stmnt num = 0.011;
     rpt line='*  SAS Version: 9.1.3';
     output;
         stmnt num = 0.012;
     rpt line='* ';
     output;
         stmnt num = 0.013;
     rpt line='*  Modifications:';
     output;
         stmnt num = 0.014;
     rpt line='*================================================================================*;';
     output;
     stmnt num = 0.015;
     rpt line ='%let chrprogram  = ' || "&chrprogram ;";   output;
     stmnt num = 0.016;
     rpt line ='%let chrprotocol = ' || "&chrprotocol ;";   output;
         stmnt num = 0.017;
     rpt line ='%let chranaltype = ' || "&chranaltype ;";   output;
         stmnt num = 0.018;
     rpt line=' '; output;
         stmnt num = 0.019;
     rpt line='      %include "/biostats/setup.sas"; ';   output;
     stmnt num = 0.020;
     rpt line=' '; output;
     stmnt num = 0.021;
     rpt line = '%let source = '||"&chrProgram/&chrProtocol/&chrAnaltype/l&&ds&i...sas;";
     output;
         stmnt num = 0.022;
         rpt_line = '%orient(landscape);';
     output;
         stmnt num = 0.023;
     rpt line='*================================================================================*;';
     output;
         stmnt num = 0.024;
      rpt line='* Generate the report                                                          ';
     output;
         stmnt num = 0.025;
      rpt line='*================================================================================*;';
     output;

         stmnt num = 0.027;
     rpt line = "  ";
     output;
```

```
%*-------------------------------------
    Generate a Title Statement
 -------------------------------------;

    stmnt num = .1;
    rpt line = "title1 " ||'" "'||";";
    output;
        stmnt num = .2;
    rpt line = "title2 "||"'"||compbl(MEMLABEL)||"- Subject Listing"||"'"||";";;
    output;
        stmnt num = .3;
    rpt line = "title3 " ||'"#PAGE"'||";";
    output;
        stmnt num = .4;
    rpt line = "title4 " ||'"  "'||";";
    output;
        stmnt num = .5;
    rpt line = "title5 "||'"&chrunder"'||";";
    output;

%*-------------------------------------
  Generate a Footnote Statement
 -------------------------------------;

    stmnt num = .6;
    rpt line = "footnote;";
    output;
        stmnt num = .7;
    rpt line = "  ";
    output;

%*-------------------------------------
  Generate the Proc Report Statement
 -------------------------------------;

    stmnt num = .8;
    rpt line = "proc report data=&libin..&&ds&i " ||
               " headline headskip spacing=1 nowindows missing;";
    output;
%*-------------------------------------
   Generate the Column Statement
 -------------------------------------;

    stmnt num = .9;
    rpt line = '     '||"columns  &colum" || ";";
    output;
      end;

%*-------------------------------------------------------------------
  Generate one Define Statement for each Var with a Statement
  Number=Var Num+1000 to insure that Define Statements follow
  Column Statements
-------------------------------------------------------------------;

    stmnt_num = 1000;

%*-------------------------------------
  If the variable is not formatted
  use Length for Width
 -------------------------------------;

   if missing(format) then
      rpt line = "define " || trim(left(name)) ||
                 " / width=" || trim(left(put(vlength,3.))) ||" left " ||
                 " '" || trim(left(label)) || "';";
       else
           do;

%*-------------------------------------------------------------------
  If the variable is formatted, but Format Length = 0 use Length for Width
  and add a Format Parameter
-------------------------------------------------------------------;

    if formatl eq 0 then
       rpt line = "define " || trim(left(name)) ||
                  " / width=" || trim(left(put(vlength,3.))) ||" left " ||
                  " format=" || trim(left(format)) ||
                  ". '" || trim(left(label)) || "';";
```

```
 %*-------------------------------------------------------------------------
   if the variable is formatted, and Format Length  use Maximum of Length
   and Format Length for Width and add a Format Parameter
 -------------------------------------------------------------------------;

        else
          rpt line = "define " || trim(left(name)) ||
                     " / width=" || trim(left(put(max(vlength,formatl),3.))) ||" left "||
                     " format=" || trim(left(format)) ||
                     trim(left(put(formatl,3.))) || ". '" || trim(left(label)) || "';";
        end;
       output;
%*-------------------------------------
 When reading the last record
 -------------------------------------;

     if eof then
       do;
            stmnt num = 9998;
         rpt line='%setft;';
                output;

%*-----------------------------------------------------------------------------
  Generate the Run Statement with a Statement Number=9999 to insure that the Run
  Statement follows all other Statements
-------------------------------------------------------------------------------;

         stmnt num = 9999;
         rpt line="run;";
                output;
                stmnt num = 10000;
         rpt_line = "  ";
       end;
run;

%*-------------------------------------
   Sort by Statement Number
 -------------------------------------;

proc sort data=list_dsn;
   by stmnt_num;
run;

%*-------------------------------------
 Generate the Ascii File
 -------------------------------------;

data  null ;
   set list dsn;
%if %length(&outdir)^=0 %then %do;
    file "&outdir.&sep.l&&ds&i...sas" RECFM=V lrecl=85 new;
    put rpt_line;
run;
%end;
    %else %do;
        file "l&&ds&i...sas" RECFM=V lrecl=85 new;
        put rpt_line;
run;
%end;
%end;
%end; ** for version;
%mend gen_list;
```