Paper 101-28

Report? Make It Easy – An Example of Creating Dynamic Reports into Excel

Liping Huang
Visiting Nurse Service of New York

Abstract

The ability to generate flexible reports in Excel is in great demand. This paper will use a real world example from a health care agency to illustrate an automated, flexible, easily managed process for intermediate and innovative SAS programmers to create updateable and individualized reports that include descriptive and statistical results. The paper will demonstrate the following capabilities by using Base SAS version 8: (1) Using %WINDOW statement to create an interface allowing end users having flexibility to define a time frame and to select specific criteria. (2) Using macro %INPUTVAR to perform various analyses by simply adding or dropping variables listed in %let statements. (3) Illustrate a procedure that adopts the features from SAS Output Delivery System (ODS) to examine the variance between two samples and select a proper probability, and subsequently export the statistically significant results to an Excel template.

Introduction

In today's health care industry, the ability to integrate comprehensive information and generate flexible reports is in great demand. An easily understandable and updateable report plays an important role in assisting health care professionals to improve quality of care and in supporting management to make better decisions.

Background

The example used for this paper is from a large urban health care agency. In order to provide higher quality of services, the agency would like to have a quarterly report but showing monthly information on adverse events occurred during care. In addition, if a report were for a specific region, the agency also would like to know if there is a statistical difference between a reporting region and the rest of the agency.

Process Illustration

Section I. Creating an Interactive Window Application

The purpose of creating a user-friendly interactive window application is to allow anyone who executes the program to have flexibility to select criteria or a time frame without editing codes. The %WINDOW REPORT shown bellow will bring a SAS AF window (see Appendix I) and store user-defined criteria as global macro variables, which are used as conditional criteria while the program performs the rest of tasks.

```
/*Assign Global Macro Variables*/

%LET begdate = ;
%LET enddate = ;
%LET region = ;
%LET user = ;
%LET password = ;

%WINDOW REPORT
 IROW= 4 ICOLUMN= 5 ROWS= 25 COLUMNS= 75
 COLOR=RED GROUP= hdrftr
 #2 @20 "Welcome to Outcome Reporting
    System"  COLOR=white
 #29 @5 "Press [ENTER]" COLOR=white
     GROUP= criteria
 #5 @5 "Please enter your user name"
    COLOR=white +3 user 9 ATTR=UNDERLINE
    AUTOSKIP=YES COLOR=white
 #7 @5 "Please enter your user password"
    COLOR=white  +3  password 9
    ATTR=UNDERLINE AUTOSKIP=YES COLOR=white
 #10 @5 "Please enter the report beginning
    date(DDMONYYYY)"  COLOR=white  +3
    begdate 9 ATTR=UNDERLINE AUTOSKIP=YES
    COLOR=white
 #12 @5 "Please enter the report ending
    date(DDMONYYYY)"  COLOR=white  +3
    enddate 9 ATTR=UNDERLINE  AUTOSKIP=YES
    COLOR=white blink
 #17 @5 "Please enter a specific region from
    the following (B, K, M, Q, S, N): "
    COLOR=white  / #20 @5 region 40
    ATTR=UNDERLINE COLOR=white ;

%MACRO REPORT;
 %DISPLAY report.hdrftr NOINPUT BLANK BELL ;
 %DISPLAY report.criteria ;
%MEND REPORT;
%REPORT;

%let borough= %bquote(')%upcase(&region)
              %bquote(');
%let beg= %bquote(')%upcase(&begdate)
          %bquote(');
%let end= %bquote(')%upcase(&enddate)
          %bquote(');
```

```
data _null_;
   call symput('cnt', put('if region = ' ||
        "&borough", $20.));
run;
%put &cnt &borough &beg &end;
```

As one can see from the log, if B is entered in the field of the region, the value of &cnt showed as: if borough = "B".

Section II. Creating data sets

The data is from the agency Oracle database. Using SQL Pass-Through facility and the global macro variables generated from %WINDOW REPORT, a data set that matches user-selected criteria is dynamically retrived from the Oracle database, named "data_all". Furthermore, a flag indicating if a patient was from a report region is also added. Data sets for a reporting region only and non-reporting regions, named "pt_region" and "pt_other" were created, respectively.

```
proc sql;
  connect to oracle (user=&user
         password=&password path=rcprod1);
  create table datain as
  select * from connection to oracle
   (select case_num, borough,
           discharge_date, adv_item from
           rc_owner.case_facts
    where discharge_date between &beg and
          &end);
  disconnect from oracle;
quit;

data data_all pt_region pt_other;
   set datain;
      &cnt then report=1;
      else report=0;
       if report>=0 then output data_all ;
          if report=1 then output pt_region;
          if report=0 then output pt_other;
run;
```

Section III. Running Analyses Based on Selected Variables

As we stated before, the purpose of this report is to analyze adverse events that occurred during care in order to help the agency monitoring the quality of patient care and making plans for improvement. Therefore, the most reporting items involving this report is related to various adverse events. In other words, in addition to overall patient information, the report will present the frequencies of adverse events within a region or across regions. Since the strategies of the agency could vary along with the time, the emphasis of individual adverse categories may vary as well. Therefore, how to minimize the efforts on modifications is the concern of programmers. By using the macro %INPUTVAR, as showed bellow, one will be able to perform various analyses by simply adding or dropping variables listed in %let statements. The %INPUTVAR is a user-defined macro. It converts variables into macro variables and calls other macros to perform various analyses, such as frequency, mean, CHISQ test and T-Test. It organizes results as a SAS data set to be output to a pre-defined Excel template.

```
/*list variables for running frequencies or
means*/

%let advfreq=adv001_1 adv001_2 adv001_3
             adv001_4 adv002_1 adv003_1
             adv004_1 adv004_2 adv005_1
             adv006_1 adv006_2 adv006_3
             adv007_1;
%let varmean=los age total_visit;
```

As mentioned before, simply add and drop variables in the above statements, one will be able to meet needs from the management. In addition, the following process shows how the program refers to the variables listed above and perform analyses by calling macros %DOFREQ, %DOMEAN, %DOCHISQ, and %DOTTEST, and organizes results into a desired order.

```
/*create data sets in order to convert  listed variables to macro
variables*/

data advfreq;
  input &advfreq;
  cards;
run;

data varmean;
  input &varmean;
  cards;
run;
```

The following %INPUTVAR converts variables into macro variables and calls other macros to perform various analyses, such as frequency, mean, CHISQ test and T-Test, and then it organizes results as a SAS data set to be output. The parameter &datain asks the data set for analysis. It could be either data for the reporting region, comparison regions, or all. The parameter &varin refers to the group of variables listed in the above %let statements, and the parameter &in is used to distinguish types of analyses. It ranges from 0 to 3 that represents performing frequency, mean, t-test, and Chisq tests, respectively.

```
%macro INPUTVAR(datain, varin, in);
 /*convert a user defined variable list from
   a %let statement to macro variables*/
   proc transpose data=&varin
        out=var2;
   run;
   proc sql noprint;
        select count(*) into :count
               from var2;
   quit;

   data _null_;
      call symput('newcount',
      compress('new') ||compress(&count));
   run;
   proc sql noprint;
     select _name_ into :new1 thru
        :&newcount from var2;
   quit;


%let count=&count;
```

```
data var3(rename=(_name_=label));
   set var2;
  obs=_n_;
run;
proc sort data=var3; by label; run;
/*calling macros to run frequencies,
 means, chisq and t-test*/
%if &in=0 %then %do;
   %DOFREQ(&count);
%end;
%if &in=1 %then %do;
   %DOMEAN(&count);
%end;
%if &in=2 %then %do;
   %DOTTEST(&count);
%end;
%if &in=3 %then %do;
   %DOCHISQ(&count);
%end;
%mend INPUTVAR;
```

Section III. Calculating Frequencies and Means

In a real world, information is usually presented as frequencies and means. The macros %DOFREQ and %DOMEAN meet this purpose. Based on the variables listed from %let statements: &DOFREQ and &DOMEAN will give results as count, percentages, and means across different months, and append results together for outputting. Please note that the macro %EXIST determines whether a base data set exists while doing appending. It deletes an existing base data set if it is the first run of a loop.

The detail codes for the %EXIST, %DOFREQ, AND %DOMEAN are listed in Appendix II.

Section IV. Do CHISQ test and T-TEST

Although it is simple enough to obtain CHISQ results as a SAS data set through the features of the PROC FREQ, it is a little tricky to have the results from a t-test organized as a SAS data set. The Base SAS version 8 has not had the capability to create a SAS data set directly from the procedure of PROC TTEST. Nevertheless, the macro %DOTTEST adopts the features from SAS Output Delivery System (ODS) and examine the variance between two samples and select a proper probability, and subsequently export statistically significant results to the Excel template (see Appendix II for detail information of %DOCHISQ and %DOTTEST.

Section V. Running Report

As stated before, the macro %INPUTVAR converts the variables listed from %let statements into macro variables and calls other macros %DOFREQ, %DOMEAN, %DOCHISQ, and %DOTTEST. For instance, by applying **%INPUTVAR**(pt_region, varfreq, 0),**%INPUTVAR**(pt_region, varmean, 1),**%INPUTVAR**(data_all, varmean, 2),and **%INPUTVAR**(data_all, varmean, 3), one will get four data sets that represent final descriptive and statistics results for a reporting region and its comparison.

Section VI. Loading Excel and output results into the template and save as a specific report

After results are calculated and organized as a SAS data set, it is time to output. There are a few different ways to load Excel template. In the example, we use 'X" command to activate an Excel template and write outputs into its "data" sheet. In addition, the codes below also write a regional title and a time frame into the data table, which will be dynamically reflected in a report title.

```
/*Load an Excel template*/
 options noxwait noxsync;
 x "c:\progra~1\micros~1\office\excel
    c:\reports\template.xls";

  data _null_;
    x = sleep(10);
  run;

/*count the total observations from the
  final data set in order to determine the
  number of rows needed to be filled in a
  data sheet of the template*/

   proc sql;
     select count(*) into :count
     from finalout;
   quit;
  /*output the final data set*/
   filename data1 dde "excel|data!r3c1:
            r%eval(&count+3)c5" notab;

    data _null_;
      set finalout;
      file data1;
      if _n_ = 1 then put 'label' '09'x
      'c_region' '09'x 'p_region' '09'x
      'c_other' '09'x 'p_other' '09'x;
       put  label $25. '09'x
            c_region 8. '09'x
            p_region 8.3 '09'x
            c_other 8. '09'x
            p_other 8.3 '09'x
            sign $8. '09'x;
    run;

/*Save the file as a specific report*/
filename cmds dde 'excel|system';
data _null_;
  archive="c:\report_&region"||".xls";;
  file cmds;
  put '[QUIT()]';
run;
```

Conclusion

This paper has shown an automated process of handling analytical results and outputting them to a pre-defined Excel template. It is easily modifiable for a variety of projects. Without require extensive knowledge of SAS, one will be able to manipulate it with ease.

References

Vyverman, K. "Using Dynamic Data Exchange to Export your SAS Data to MS Excel". *Proceedings of the twenty-seventh Annual SAS Users Group International Conference, paper 5, 2002.*

Abdurazak, T. "Using SAS Macros to Create Automated Excel Reports Containing Tables, Charts and Graphs". *Proceedings of the twenty-seventh Annual SAS Users Group International Conference, paper 126, 2002.*

Mace, M. "%Window: You Can Talk to the Users, and They Can Talk Back". *Proceedings of the twenty-seventh Annual SAS Users Group International Conference, paper 192, 2002.*

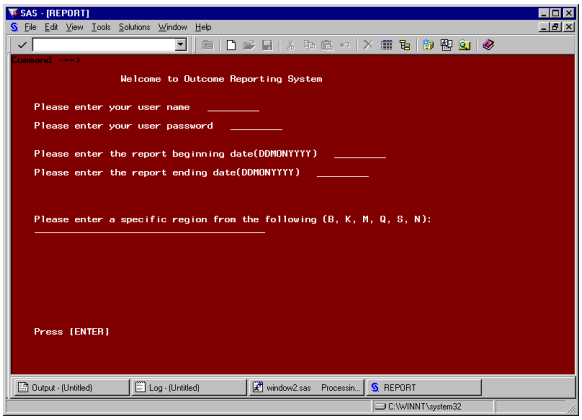
Trademarks

Contact Information

Liping Huang
Center for Home Care Policy and Research
Visiting Nurse Service of NY
5 Penn Plaza, 11th Floor
New York, NY 10001
(212) 290 – 0520
E-mail: liping.huang@vnsny.org

APPENDIX I

SAS AF WINDOW: Generated from %REPORT



APPENDIX II

```
%macro EXIST(dsn);
   %if %sysfunc(exist(&dsn)) %then %do;
        proc datasets library=work;
             delete &dsn; quit;
   %end;
%mend EXIST;


%macro DOFREQ(count);
  %do i=1 %to &count;
    %let n=&i;
     proc freq data=&datain noprint ;
         tables month*&&new&n /list out=out;
     run; /*&&new&n was created from
            %inputvar*/
    data out;
      set out;
        if &&new&n >0;/*Only keep real
                        value*/
    proc transpose data=out
         (keep=count month)
         out=outa(drop=_label_ _name_)
         prefix=c_m;
         id month;
    proc transpose data=out(keep=percent
         month)
         out=outb(drop=_label_ _name_)
         prefix=p_m;
         id month;


   /*merge count and percent together*/
   data out&n;
       length label $25.;
         merge outa outb;
         label="&&new&n";
   run;
   /*delete base data set if it is the
     first run*/
   %if &n=1 %then %do;
       %exist(freq);
   proc append data=out&n(rename=(&&new&n
                   =value))
        base=freq force;
   run;
  %end;
 %else %do;
   proc append data=out&n(rename=(&&new&n
     =value))
     base=freq force;
```

```
   run;
 %end;
%end;
 /*organize data into a desired order*/
  proc sort data=freq; by label; run;
  data freq(drop=value);
       merge var3 freq;
 /*var3 is a data set created from
  %inputvar, with a desired order*/
       by label;
  run;
  proc sort data=freq; by obs; run;
%mend DOFREQ;


%macro DOMEAN(count);
 %do i=1 %to &count;
    %let n=&i;
  proc means data=&datain nway noprint ;
       class month;
       var &&new&n;
      output out=out(keep=qtr count
            mean)n=count
            mean=mean/autoname;
 run;
 /*separated count and means*/
 proc transpose data=out(keep=count
     month)
     out=outa(drop=_label_ _name_)
     prefix=c_m;
     id month;
 run;
 proc transpose data=out(keep=mean
     month)
     out=outb(drop=_label_ _name_)
     prefix=p_m;
     id month;
 run;
/*combine outputs for count and mean together*/
 data out&n;
    length label $25.;
    merge outa outb;
      label="&&new&n";
 run;
/*appending to base file*/
%if &n=1 %then %do; %exist(mean);
proc append data=mean&n
       base=mean force;
run;
%end;
%else %do;
  proc append data=mean&n base=mean
       force;
  run;
%end;
%end;
  proc sort data=mean; by label; run;
  data mean;
    merge var3 mean;
    by label;
  run;
 proc sort data=mean;
   by obs; run;
%mend DOMEAN;


%macro DOCHISQ(count);
  %do i=1 %to &count;
  %let n=&i;

/*perform a CHISQ tests between reporting
  region and comparison group*/

  proc freq data=&datain noprint;
       tables report*&&new&n / chisq;
       output out=out(keep=p_pchi)chisq;
  run;

  data out;
     length label $20.;
     set out;
     label="&&new&n";
  run;
 /*delete the base file is it is the f
  run of appending*/
  %if &n=1 %then %do;
    %exist(chisq);
     proc append data=out base=chisq force;
          run;
  %end;

  %else %do;
     proc append data=out
        base=chisq force;
     run;
  %end;
 %end;
    proc sort data=chisq;
       by label;
     run;

/*making statistical judgement and organize
  results to be output*/

   data chisq(keep=label sign);
      merge var3 chisq;
       by label;
         if p_pchi <= 0.001 then
             sign='***';
           else if 0.001 < p_pchi <
                0.01 then sign= '**';
              else if 0.01 < p_pchi
                <= 0.05 then sign= '*';
   run;
   proc sort data=chisq;
        by obs; run;
%mend DOCHISQ;


%macro DOTTEST(count);
  %do i=1 %to &count;
      %let n=&i;
  /*using ODS features to store results
    from t-test*/
   ods output ttests=out1(keep=variable
       variances tvalue probt);
   ods output Equality=out2
       (keep=variable fvalue probf);
   ods trace on;
   proc ttest data=&datain noprint;
       class report; var &&new&n;
   run;
   ods trace off;
 /*using variance to measure group
   equalience*/
   data out2(drop=variable)
     length label $20.;
       merge out1 out2;
         by variable;
       if probf <=0.05 then do;
         if variances='Unequal';
       /*output results with unequal
         variance*/
        end;
        else do;
         if variances='Equal';
       /*output results with equal
         variance*/
        end;
        label = " &&new&n"
```

```
     run;
    %if &n=1 %then %do;
      %exist(test);
       proc append data=out2 base=test force;
         run;
    %end;
    %else %do;
       proc append data=out2 base=test force;
         run;
    %end;
  %end;

/*create a final result set from ttest and
organized it into a desired order*/

     proc sort data=test; by label;run;
     proc sort data=var3; by label; run;

     data ttest(keep=label sign);
       merge var3 test;
         by label;
          if probt <= 0.001 then sign='***';
          else if 0.001 <= probt < 0.01
            then sign= '**';
            else if 0.01 <=probt < 0.05
               then sign= '*';
    run;

     proc sort data=ttest; by obs; run;

%mend DOTTEST;
```