Creating Dynamic Reports Using Data Exchange to Excel

Liping Huang
Visiting Nurse Service of New York

ABSTRACT

The ability to generate flexible reports in Excel is in great demand. This paper illustrates an automated and easily managed process to calculate basic elements of a report, such as frequencies and means. The process also performs some basic statistical analysis to organize results and automatically generate a report based on a pre-designed 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 up to date report plays an important role in assisting health care professionals to improve quality of care and in supporting management to make better decisions.

Generating reports involves organizing streamlined data, analyzing data, and presenting user-friendly results. In other words, it makes information useful in the real world. However, making the process to be more efficient and less labor intensive is a challenge for programmers and analysts. This paper uses a real world example to provide a detailed walkthrough of how to programmatically create and customize an Excel workbook containing SAS data.

Background

The example used for this paper is from a large urban health care agency. In order to provide higher quality services, the agency would like to have reports on patients and the agency administration on a regular basis. Providing up to date reports in a user-friendly format is a major responsibility of the programmers.

Process Illustration

Section I. Excel Template Preparation

Because of the popularity and user-friendly interface of Excel, reports created by Excel are preferred by management. The first, and most important task, is to design and prepare a report template. More often, it is the most time-consuming stage. Programmers must not only have technical knowledge, they must also fully understand the purpose of a report. Enhanced communication skills will always help to achieve this goal.

The example used in this paper contains multiple sheets with tables, charts and graphics. In addition, a sheet, named “data”, is used to receive analytical results from SAS. All results presented in the report are linked to the “data” sheet.

Section II. Creating an Interactive Window Application

The purpose of creating a user-friendly interactive window application is to allow anyone who executes program to have the flexibility to select criteria or a time frame without having to edit codes. Instead of using %let statement, the %WINDOW REPORT shown bellow will store user-defined criteria as global macro variables, which are used as conditional criteria while the program processes the rest of the procedures.

%WINDOW REPORT
  IROW= 4 ICOLUMN= 5 ROWS= 25 COLUMNS= 75
  COLOR=GRAY
  GROUP=hdrftr
    #2 @20 "Welcome to Outcome Reporting Systeme" COLOR=BLUE
    #20 @5 "Press [ENTER]"
  GROUP=criteria
    #5 @5 "Please enter the report beginning date (DDMONYYYY)" +3 begndate 9
    ATTR=UNDERLINE AUTOSKIP=YES COLOR=RED
    #6 @5 "Please enter the report ending date (DDMONYYYY)" +3 enddate 9
    ATTR=UNDERLINE AUTOSKIP=YES COLOR=RED
    #11 @5 "Please enter a specific region from the following: B, K, M, Q, S, N" / @ region 40
    ATTR=UNDERLINE color=RED;

%LET begndate = ;
%LET enddate = ;
%LET region = ;
%MACRO REPORT;
  %DISPLAY report.hdrftr NOINPUT BLANK BELL ;
  %DISPLAY report.criteria ;
  %END REPORT;
  %REPORT;
After the parameter values are defined, a conditional statement is created from the following codes:

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

As you can see from the log, if you entered B in the field of the region, the value of &cnt showed as: if borough = “B”.

**Section III. Creating data sets**

The macro variable &cnt is used to create two subsequent data sets for calculating frequencies and means, as well as to indicate whether a patient resides in a report region or in other regions.

```plaintext  
%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 &&new&n/list out=out&n;  
run;  
data out&n;  
length label $25.;  
set out&n;  
if &&new&n>0;  
label="&&new&n";  
run;  
%if &n=1 %then %do;  
%exist(freq);  
proc appenddata=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;  
%mend DOFREQ;  
%macro DOMEAN(count);  
%do i=1 %to &count;  
%let n=&i;  
proc means data=&datain noprint;  
var &&new&n;  
output out=out&n mean=mean;  
run;  
data mean&n;  
length label $15.;  
set out&n(drop=_type);  
```

The data sets pt_region and pt_other are created for the purposes of calculating frequencies and means for a reporting region and other regions, respectively. The data set data_all is created for the purpose of performing statistical comparisons between a reporting region and other regions.

**Section IV. Calculating Frequencies and Means**

In the real world, information is usually presented as frequencies and means. The macros %DOFREQ and %DOMEAN meet this purpose. Based on the items put in separate %let statements: &varfreq and &varmean, they calculate frequencies and means and append results together for outputting. Also note that the macro %EXIST determines whether a data set exists while doing appending. It deletes an existing base data file if it is the first run of a loop.

The following is the codes of macros %EXIST, %DOFREQ, AND %DOMEAN.

```plaintext  
/*the macro for deleting an existing file*/  
%macro EXIST(dsn);  
      %if %sysfunc(exist(&dsn)) %then %do;  
      proc datasets library=work;  
      delete &dsn; quit;  
    %end;  
%mend EXIST;  
/*the macro for calculating frequencies*/  
%macro DOFREQ(count);  
      %do i=1 %to &count;  
      %let n=&i;  
      proc freq data=&datain noprint;  
      tables &&new&n/list out=out&n;  
      run;  
      data out&n;  
      length label $25.;  
      set out&n;  
      if &&new&n>0;  
      label="&&new&n";  
      run;  
      %if &n=1 %then %do;  
      %exist(freq);  
      proc appenddata=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;  
      %mend DOFREQ;  
/*the macro for calculating means*/  
%macro DOMEAN(count);  
      %do i=1 %to &count;  
      %let n=&i;  
      proc means data=&datain noprint;  
      var &&new&n;  
      output out=out&n mean=mean;  
      run;  
      data mean&n;  
      length label $15.;  
      set out&n(drop=_type);  
```

The following is the codes of macros %EXIST, %DOFREQ, AND %DOMEAN.
Section V. CHISQs and T-TESTs

In order to help regional managers compare patients residing in one region with patients residing in other regions, CHISQ tests for all binary variables and T-TESTS for all continuous variables are performed. 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. Version 8 SAS does not have the capability to create a SAS data set directly from the results of PROC TTEST. Nevertheless, the macro %DOTTEST adopts the features from SAS Output Delivery System (ODS) and examines the variances of a sample to determine a sample equality or inequality, and therefore to define a t-value and its probability.

```
%macro DOTE1(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;
  %end;
%mend DOTE1;
```

/*the macro running t-test for listed continuous variables*/

```
%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;
  %end;
%mend DOTTEST;
```

/*the macro for calculating chisq values for listed binary variables*/

```
%macro DOCHISQ(count);
  %do i=1 %to &count;
    %let n=&i;
    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;
  %end;
%mend DOCHISQ;
```
Section VI. Running %DOFREQ, %DOMEAN, %DOCHISQ, and %DOTTEST

The macro %INPUTVAR converts the variables listed from %put statements into macro variables and calls other macros %DOFREQ, %DOMEAN, %DOCHISQ, and %DOTTEST for generating a final output data set that includes the frequencies and means for a reporting region and other regions, as well as the statistical difference between them. The following codes shows how to put everything together to generate a final SAS data set.

/*list variables for running frequencies or means*/
%let socfreq=agegroup1854 agegroup5564 agegroup7584 agegroup85 white black hispanic asian otherrace female male nocharge duallyeligible medicareonly medicaidonly anyhmo privateinsurance otherpayment;
%let varmean=los visit_ns;

/*create data sets in order to convert listed variables to macro variables*/
data varfreq;
input &socfreq;
cards;
runk;
data varmean;
input &varmean;
cards;
runk;

%macro INPUTVAR(datain, varin, in);
/*convert listed variables 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;

/*assign observation numbers in order to merge*/
data var3(rename=(_name_=label));
set var2;
obs=_n_; run;
proc sort data=var3; by label; run;

/*run frequencies, means, chisq and t-values*/
%if &in=0 %then %do;
%DOFREQ(&count);
data freq(drop=value);
merge var3 freq by label; run;
proc sort data=freq; by obs; run;
%end;
%if &in=1 %then %do;
%DOMEAN(&count);
data mean(rename=(freq=count mean=percent));
merge var3 mean by label; run;
proc sort data=mean; by obs; run;
%end;
%if &in=2 %then %do;
%DOTTEST(&count);
data ttest;
merge var3 ttest 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='*';
runk;
end;
data=ttest;
by obs;
run;

%end;
%if &i n=3 %then %do;
%DOCHISQ(&count);
data chisq;
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;
%end;
%mend INPUTVAR;

/*create an output data set for the reporting region*/
%inputvar (pt_region, VARFREQ, 0);
%inputvar (pt_region, VARMEAN, 1);
data region(rename=(count=c_region percent=p_region));
set freq mean ;
run;

/*create an output data set for the comparison*/
%inputvar (pt_other, VARFREQ, 0);
%inputvar (pt_other, VARMEAN, 1);
data other(rename=(count=c_other percent=p_other));
set freq mean ;
run;

/*examine statistical significance between the reporting region and the comparison*/
%inputvar (data_all, varmean, 2);
%inputvar (data_all, varfreq, 3);

/*combine outputs together*/
data output;
merge region other;
by label;
run;

/*modify the data set for simplifying the output from the chisq and ttest results*/
data significant(keep=label sign);
set chisq ttest;
run;

/*the final output data set*/
data finalout;
merge output significant;
by label;
run;

**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 the 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;

/*write a report title and the time frame into a data sheet*/
filename data1 dde "excel!data!r1c1:r1c6" notab;
data _null_;
file data1;
title=compress("&region");
beg=compress(year("&beg"d))||''||compress(month("&beg"d));
end=compress(year("&end"d))||''||compress(month("&end"d));
put title $25. '09'x
beg $10. '09'x
end $10. '09'x;
run;

/*count the total observation from the final output 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;

Finally, the results are ready to be outputted. Using the most popular and powerful protocols for integrating SAS and Excel - Dynamic Data Exchange (DDE), this code will output the desired results to the template and save them as a particular regional report.

/* 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;

FILENAME CMDs DDE 'excel|system';

DATA _NULL_;  
Archivo="c:\report_&boro\".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 having much knowledge of SAS, anyone who runs the program will be able to manipulate it with ease.

References


Trademarks

SAS is a registered trademark of SAS Institute Inc. in the USA and other countries. ® Indicates USA registration.

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

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