PharmaSUG 2014 - Paper CC33

QC made easy using macros PharmaSUG 2014

Author1 Prashanthi Selvakumar, Percept Pharma Services, Bridgewater, NJ

ABSTRACT

To err is human. But this cannot be an excuse for our mistakes. We have to make sure there are no errors in the reports that we submit. To minimize errors in programming, we have the QC (Quality Check) team. In most cases, QC requires producing reports from the scratch, which is time consuming. In this paper we will be discussing a macro used for creating reports similar to the production programmer and that compares the result with QC programmer. This can help you save time on coding as well check the mistakes that human eyes do not capture.

INTRODUCTION TO QUALITY CONTROL

Quality control refers to the process of maintaining the standards by verifying the output. For a statistical programmer, the outputs will include tables, listings figures or datasets. For programming using clinical trial data, quality control or QC in short, refers to ensuring, that the output conforms to the specification. The quality control involves three steps they are (Gorrell, 2003)

1) QC of input data

2) QC of output information

The figure 1.1 describes how the output is produced from the Case report form (CRF) information collected at clinical trials sites. The second row describes the different stages QC including the checks in the dataset and final output.

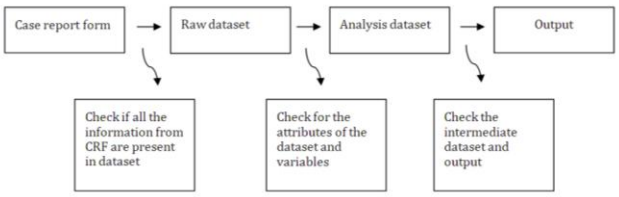

Figure1.1 The flow of information and Quality checks from Case report form to output

This paper will describe in detail, the quality checks to be done before submitting the output.

QUALITY CONTROL OF INPUT DATASET

To represent all the information collected in CRF

Case Report Form (CRF) is the information that is collected at the subject level and it is transferred from the clinical trial sites, lab sites to databases and then datasets. There are separate CRF pages for each dataset, like lab, adverse events, demographic information etc.

The lab CRF page, gives information like the labtest name (Hemoglobin, Creatinine clearance) and labtest codes, in addition to other variables like (protocol number, subject ID and others). The following SAS code, compares if all the lab parameters collected in the CRF are present in the dataset.

```
PROC SQL;

   CREATE TABLE tocheck AS   /*tocheck is the table created for the missing lab
parameters collected in the CRF but missing in the raw data*/

   SELECT DISTINCT lbtest, labcode

   FROM crf /* crf is the SAS dataset from CRF page*/

   WHERE lbtest NOT IN
```

```
      (SELECT lbtest from lab_safe);
QUIT;
```

Similar to the above code, we can compare if all the information that is collected in the CRF is represented in the raw dataset (code not shown). This is very important as this will ensure, all the parameters in the dataset are mapped to the corresponding fields in the dataset.

To check the attributes of dataset and variables

The next step, is to determine if the input dataset exists. After checking for the existence of the dataset, we have to check the sorting order, version of SAS and other attributes of the dataset. Then the numeric and character variable attributes are checked. We have to verify the length of variables, format, and extreme values of numeric variables. In addition, we have to make a note of all the observations with missing values for all variables and the number of missing values for both numeric and character variables.

Once the pre- processing is complete, other specific and detailed analysis can be done on the dataset. Based upon the information that is collected, this macro can be used across all the datasets.

```
/*-------------------------------------------------------------------------------*/
SECTION I – Checking the attributes of the dataset and variables
STEP1 - to check for the existence of the dataset, if the dataset exists then proceed
to step2 – 4, if it does not exist then exit
STEP2 - if the dataset exists then open the dataset and then count the num of
variables and observation
STEP3 - to count the number of observations with missing values for all variables
/*-------------------------------------------------------------------------------*/
OPTIONS MLOGIC MPRINT SYMBOLGEN;
/* to check for the existence, counting the variables and obs of the dataset*/
%MACRO check(dsn);
%GLOBAL nobs nvars miss ;
   %IF %SYSFUNC(exist(&dsn))%THEN %DO;/*STEP1*/
        /*this is to open the dataset, count the num of obs,vars and then close*/
        DATA _null_;
          FILE print;
          PUT #1 "Data set &dsn. exists"; /*to check if dataset exists*/
          %LET dn= %SYSFUNC(OPEN(&dsn)); /*STEP2*/
          %IF &dn %THEN %DO;
             %LET nobs =%SYSFUNC(ATTRN(&dn,nobs)); /*to count the num. of obs.*/
             %LET nvars=%SYSFUNC(ATTRN(&dn,nvars)); /*to count the num. of vars.*/
             %LET rc =  %SYSFUNC(CLOSE(&dn));/*to close the dataset*/
             DATA _null_;
               FILE print;
               PUT # 1  "and has &nvars variables and &nobs";/*this prints output*/
             DATA _null_; /*STEP3 - to check if all the values of an obs are missing*/
               SET &dsn;
               %LET miss= %SYSFUNC(CMISS (of _all_));/*if all the values are missing*/
               %IF &miss GT 0 %THEN %DO;
                  PUT #1 "The number of observations with all missing values= &miss";
```

```
            %END;
          %END;
        %END;
   %ELSE %DO;
       DATA _null_; /*STEP1 - if the dataset does not exist*/
          FILE PRINT;
          PUT #3 @10 "Data set &dsn. does not exist";
       RUN;
   %END;
%MEND check;
%CHECK (lab_safe);
```

Detailed analysis of the variables

Using the PROC SQL and INTO clause, macro variables are generated (CLIST, CHARCT). The macro variable Charct resolves to 34, which is the total number of character variables in the dataset.

This CHARCT is then used in the next step where each of the character variable is converted into the macro variable CLIST1, CLIST2, CLIST3, CLIST34. The value of these macro variables will be, labname, protno, labtest, and other character variables. Similarly, we create a macro variable called (req), using positional parameters. This is invoked using values like protno,subjid and others. In the same way, we can count the missing, non- missing , mean, extreme observation for numeric variables (code not shown).

```
/*-------------------------------------------------------------------------------*/
SECTION II - Detail analysis of character and numeric variables
STEP 1 Select the name of the character variables and create one single macro variable
'CLIST' where the variable names are separated by a space - " "
STEP 2 To count the num. of character variables and then store it in a macro variables
STEP 3 To use this variable &CHARLIST to produce the frequency for all the variables
with missing required variables - &req that is discussed later
STEP 4 To create a macro variables - &req (required character variable) using
positional parameters and later invoking the macro.
/*-------------------------------------------------------------------------------*/
%MACRO VAR (req=); /*STEP 4*/
%LET lib=WORK;
%LET dsn=LAB_SAFE; /*change the name of the dataset and the library name if needed*/
%GLOBAL charct numct; /*we have to use this macro variable in the next steps*/
PROC SQL NOPRINT; /*STEP1*/
   SELECT name, PUT(COUNT(name),5.-L) INTO:clist separated by ' ' , :charct
   FROM dictionary.columns /*STEP 1 AND STEP 2*/
   WHERE LIBNAME=UPCASE("&lib") and MEMNAME=UPCASE("&dsn") and TYPE='char';
   %PUT "total no. of character varibles = &charct"; /*this will print the count as a
macro variable in the output*/
QUIT;
PROC SQL NOPRINT ; /*STEP 2*/
   SELECT name INTO:clist1 - :clist%LEFT(&charct)/*charct is created in step 2a*/
   FROM dictionary.columns
```

```
   WHERE LIBNAME=UPCASE("&lib") and MEMNAME=UPCASE("&dsn") and TYPE='char';
QUIT;
RUN;
PROC FREQ DATA = &dsn; /*STEP 3A*/
   TABLES &clist;  /*clist is created in step 1 above*/
   WHERE &req = " ";
   TITLE "the list of all the character variables with missing &req";
RUN;
%MEND var ;
%var (req= protno);
%var (req=subjid);    /*STEP 4 - calling the macro - VAR with required variables*/
/*for more details regarding the PROC SQL and INTO CLAUSE please visit the following
website http://support.sas.com/kb/24/612.html */
```

QUALITY CONTROL OF THE OUTPUT INFORMATION

To create (N) number of similar reports using ONE macro

The output information in a clinical trial consists of tables, listings and figures, that are submitted according to CDISC or company standards. The list of tables (LOT) consists of all the list of tables, listings, reports and adhoc reports that have to be created, along with the title of the table, the lab parameters used for that particular report, and other important information.

In one of the study, we had to create 34 descriptive statistics tables. After sub setting the reports with similar template, we have to create 34 different reports for lab parameters (hemoglobin, creatinine, aspartate amino transferase). Instead of writing (N) different programs a single macro can create all the reports and store it with the name of the table in the LOT as (N) different reports.

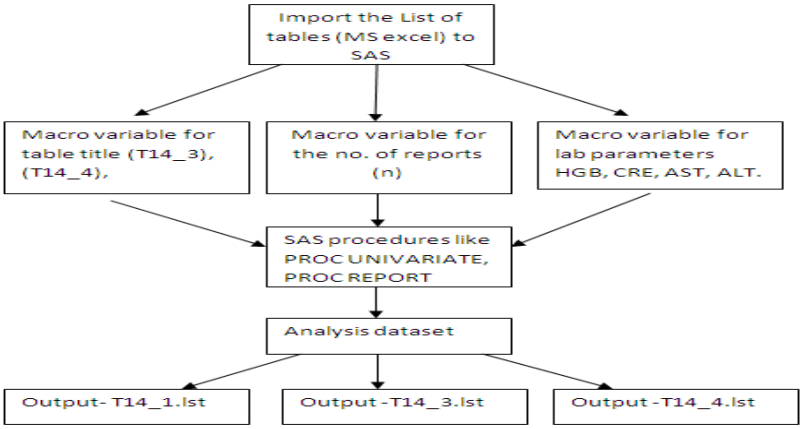

Figure1.2 Description of the report macro – to produce multiple reports

As described in the figure 1.2, we start with the import of List of tables, which has information about the table number, title, dataset and the lab parameter that is used in the creation of tables. After importing the LOT into SAS, different macro variables are created for lab parameters, number of reports and title of the reports to be generated.

The following SAS code is used for generating multiple reports and storing them with their respective names from the LOT.

```
/*-------------------------------------------------------------------------------*/
SECTION IV - Creation of multiple reports
```

STEP 1
A) First the List of all the tables (LOT) to be produced is imported to SAS from excel
B) The List of tables (LOT) is then sub-setted based upon the required tables to be generated

STEP 2
A) It counts the number of efficacy parameters (like HGB,CRE,CRE_CL) and it creates one macro variable called n

B) Creates another macro variables called (var1 - var34) which has the table title
eg. var1 = T14_3, var2= T14_2, var3= T14_5 ……..
The values of the macro variables var1 - var34 depends upon the LOT (list of table titles)
C) Then the macro variables ef1 - ef34 are created based upon the LOT (the list of the parameters) eg. ef1 =HGB ef2= HGB ef3= CRE and so on.
These values are used to calculate the descriptive stats. in the next steps

STEP 3
A) The report macro where the input dataset is the lab_v and it created different datasets based upon the parameters mentioned above
B) The analysis variables for the desc. stats are either the standard results or the change from baseline and percent change from baseline - avarC) The parameter that changes is the n, which is the number of the reports to be generated that depends on the LOT – n

D) After this the report generated it is stored in the same location in UNIX, using the &var1.lst -- &var34.lst, when they resolve they give the name of the table in the LOT as mentioned above.

```
/*-------------------------------------------------------------------------------*/

/*STEP 1 – the LOT is imported into SAS – code not shown here*/

%GLOBAL n;

PROC SQL ; /*STEP 2A,B,C*/

   SELECT PUT (COUNT(efques),3.), tt,efques  INTO :n, :var1 - :var%left(&n),:ef1 -
:ef%left(&n)  /*STEP 2*/

   FROM lot /*this the list of tables to be generated for the study*/

   QUIT ;

RUN;

%MACRO report(no=,avar=); /*STEP 3B,3C - creating &n and &avar*/

DATA &&ef&no;

   SET perm.lab_v; /*STEP 3A this input dataset can change - if changes, change this
to &dsn and invoke at the end*/

   WHERE efques= "&&ef&no" and 0 LE viswin LE 12;

RUN;

DATA trt&no fas&no;

   SET &&ef&no;

   IF treated =1 THEN OUTPUT  trt&no;

   IF fasflagn = 1 THEN OUTPUT fas&no;

RUN;

DATA trt&no;

   SET trt&no;

   IF ptrand NE . and factdat NE .;

   IF random=1 and treated =1;
```

```
RUN;
PROC SORT DATA = trt&no;
    BY viswin xviswin treattxt;
RUN;
PROC UNIVARIATE NOPRINT DATA = trt&no;
   VAR &avar; /*STEP 3B invoked at the last */
   OUTPUT OUT= &&ef&no N= n MEAN=mean STD= std MIN= min Q1= q1 MEDIAN=median Q3=q3
max=max;
   BY viswin xviswin treattxt;
RUN;
ODS LISTING ; /*STEP 3D */
ODS LISTING FILE="../analysis/&&var&no...lst"; /*the macro variable var1 - var34 with
titles is used to save the output*/
PROC REPORT DATA=&&ef&no  HEADLINE HEADSKIP NOWD SPLIT= "*" SPACING=2 ;
        COLUMNS xviswin treattxt n mean std min q1 median q3 max ;
        DEFINE xviswin/ORDER ORDER = DATA "Visit" WIDTH=15 ID LEFT;
        DEFINE treattxt/DISPLAY "Treatment" WIDTH= 30 LEFT;
        DEFINE n/DISPLAY "N" WIDTH=3 CENTER FORMAT=3.;
        DEFINE mean/DISPLAY "MEAN" WIDTH=8 CENTER FORMAT=7.2;
        DEFINE std/DISPLAY "ST. DEV" WIDTH=8 CENTER FORMAT=7.2;
        DEFINE min/DISPLAY "MIN" WIDTH=8 CENTER FORMAT=7.2;
        DEFINE q1/DISPLAY "Q1" WIDTH=8 CENTER FORMAT=7.2;
        DEFINE median/DISPLAY "MEDIAN" WIDTH=8 CENTER FORMAT=7.2;
        DEFINE q3/DISPLAY "Q3" WIDTH=8 CENTER FORMAT=7.2;
        DEFINE max/ DISPLAY "MAX" WIDTH=8 CENTER FORMAT=7.2;
        BREAK AFTER xviswin/SKIP;
RUN;
 ODS LISTING CLOSE;
```

This is the sample code that describes the Fig1.2 in detail. If we want to save the log file, we can use the same macro variable (&&var&no) to save your log (PROC PRINTTO – code not discussed). If we have to calculate the descriptive statistics for any efficacy datasets, then you can change the input dataset to a macro variable and invoke it later. Thus several (n) number of reports are produced using this one macro, which can be modified to produce many more. We can include the Confidence intervals

OTHER TIPS

Autoexec.sas

The Autoexec.sas is a file that is located in the same directory as the SAS.exe. Every time we start the SAS session this file runs. There are some SAS statements that we commonly use in one protocol/ project like:

1) Libname statement
2) Options NOFMTERR, MLOGIC, SYMBOLGEN,
3) Format library

When we add these lines to the autoexec.sas, you do not have to type them and re run them, every time you start a SAS session.

Dynamic data exchange

If you wanted to write comments to the LOT like QC from dataset or from comparing the reports, we can use the Dynamic data exchange (DDE). This enables you to write a line, comment to the excel file directly. This is not discussed, but can be used with this program.

CONCLUSION

Thus in this paper we have described the process of verifying the input dataset and how it is made simple using macros. It also describes a macro, to check the numeric and character variable. Finally, macros can be used efficiently to create several reports, as mentioned in the list of tables. As a QC programmer, even though the work assignment is the same as the production programmer, the QC programmer does not have access to the standard macros and, or codes like the production programmer. Creation of these QC macros will help save time on QC and also ensure that the reports submitted are accurate.

REFERENCES

Carpenter, Art, 2004, Carpenter's Complete Guide to the SAS® Macro Language 2nd Edition, Cary, NC: SAS Institute Inc.,2004.

Carpenter, Arthur L., 1997, Resolving and Using &&var&i Macro Variables, presented at the 22nd SAS User's Group International, SUGI, meetings (March, 1997) and published in the Proceedings of the Twenty-Second Annual SUGI Conference, 1997.

Gorrell, Paul>. 2003. “Quality Control with SAS numeric data.” *Proceedings of the NESUG 2003 Conference*. Cary, North Carolina: SAS Institute Inc Available at http://www.nesug.org/Proceedings/nesug03/at/at001.pdf

Yindra, Chris. 2003. “%SYSFUNC - The Brave New Macro World.”*Proceedings of the SUGI23 1998 Conference*. Cary, North Carolina: SAS Institute Inc Available at http://www2.sas.com/proceedings/sugi23/Advtutor/p44.pdf

Carpenter, Arthur. 2003. Storing and Using list of values in a macro variable.” *Proceedings of the SUGI29 2004 Conference*. Cary, North Carolina: SAS Institute Inc Available at http://www.lexjansen.com/pnwsug/2004/c_cc_storing_and_using_a_lis.pdf

SAS Institute inc. “Delete variables that have only missing values. 2014. Available at http://support.sas.com/kb/24/612.html.

SAS Institute inc. “Creating and using macro variables.” 2014. Available at http://support.sas.com/documentation/cdl/en/sqlproc/63043/HTML/default/viewer.htm#p0lsf4btafkw9mn1md4c69kkfwbl.htm

ACKNOWLEDMENTS

Percept Pharma Services
1031 US highway
Bridgewater, NJ 08807

Eliassen Group Biometrics and Data Solutions
351, N Frontage road,
New London, CT 06320

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:
Enterprise: Percept Pharma Services
E-mail: prashanthiselvakumar@gmail.com