Paper 7100-2016

3N Validation to Validate PROC COMPARE Output

Amarnath Vijayarangan, Emmes Services Pvt Ltd, India

ABSTRACT

In the clinical research world, data accuracy plays a significant role in delivering quality results. Various validation methods are available to confirm data accuracy. Of these, double programming is the most highly recommended and commonly used method to demonstrate a perfect match between production and validation output. PROC COMPARE is one of the SAS® procedures used to compare two data sets and confirm the accuracy. In the current practice, whenever a program rerun happens, the programmer must manually review the output file from the PROC COMPARE to ensure an exact match. This is tedious, time-consuming, and error prone because there are more output files to be reviewed and manual intervention is required. The proposed approach programmatically validates the output of PROC COMPARE in all the programs and generates an HTML output file with a Pass/Fail status flag for each output file. The status is flagged as Pass whenever the output file meets the following 3N criteria:

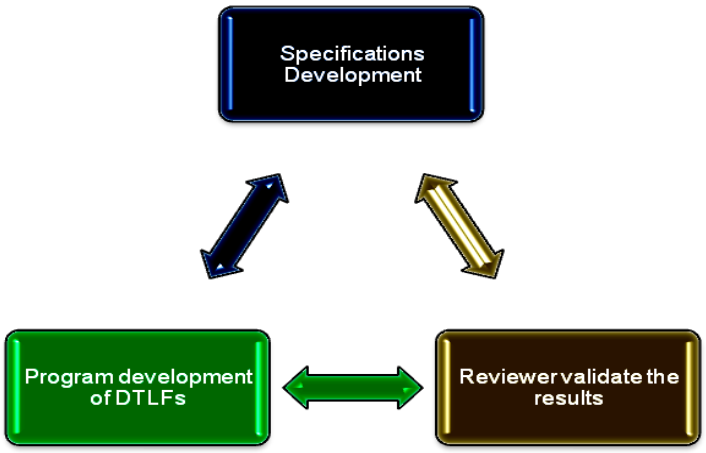
1. **N**OTE: No unequal values were found. All values compared are exactly equal.
2. **N**umber of observations in base and compared data sets are equal.
3. **N**umber of variables in base and compared data sets are equal.

INTRODUCTION

Every clinical study involves numerous data sets, tables, listings and graphs to validate and it is very crucial for the reason that these data characterize the subjects in the clinical study. Presently, the output of Tables (T), Listings (L) and Figures (F) are also available as data sets to speed up the validation process. These primary program output data sets are compared with the QC output data sets USING PROC COMPARE to ensure the accuracy. As of now, whenever the program rerun happens, programmer must manually review every output file for the statement "NOTE: No unequal values were found. All values compared are exactly equal". If this statement is present in the output file, it is concluded that two data sets are identical. Unfortunately, this assumption is not always true. Whether it is true or not, it is cumbersome task and time consuming to go through each and every output file when the number of programs or rerun increases and also error prone due to manual intervention. The approach proposed in this paper explains how to programmatically validate the output generated from PROC COMPARE of all the programs and also it ensures the accuracy.

TYPICAL PROGRAM LIFE CYCLE

A typical work day of a programmer starts with Data sets (D), Tables, Listings and Figures. The process flow and the role of primary programmers, review programmers and statisticians as follows:



Display 1. Program Life Cycle

If there are any validation comments raised by the reviewer, both the primary and review programmers should first review the findings together and resolve the issues. Still if it is not resolved they discuss with statistician to come to a conclusion.

VALIDATION USING PROC COMPARE

PROC COMPARE is one of the widely used validation procedure to compare two data sets. The basic syntax is as follows:

```
proc compare base=qcdata compare=datatovalidate;      run;
```

To expedite the validation process, the data sets that are used to generate the production DTLFs are saved as a permanent data set and it is being compared with the validation output data set developed by the reviewer using the above mentioned code.

The typical output generated by PROC COMPARE will have four sections namely data set summary, variable summary, observation summary and optionally compare differences if there is any discrepancies found between the data sets. The programmer looks for the statement “NOTE: No unequal values were found. All values compared are exactly equal.” in the PROC COMPARE output as follows:

```
proc compare base=sashelp.class compare=sashelp.class;      run;
```

```
                                Data Set Summary

Dataset                 Created          Modified  NVar    NObs  Label

SASHELP.CLASS   24MAY11:13:52:29  24MAY11:13:52:29     5      19  Student Data
SASHELP.CLASS   24MAY11:13:52:29  24MAY11:13:52:29     5      19  Student Data

                              Variables Summary

                    Number of Variables in Common: 5.

                             Observation Summary

                         Observation      Base  Compare

                         First Obs           1        1
                         Last  Obs          19       19

        Number of Observations in Common: 19.
        Total Number of Observations Read from SASHELP.CLASS: 19.
        Total Number of Observations Read from SASHELP.CLASS: 19.

        Number of Observations with Some Compared Variables Unequal: 0.
        Number of Observations with All Compared Variables Equal: 19.

        NOTE: No unequal values were found. All values compared are exactly equal.
```

Display 2: Proc Compare Expected Output

But the presence of this statement does not always guarantee the accuracy of comparison. The PROC COMPARE can be misleading you in its output if the user is not very cautious. The below sample code compares the same data sets sashelp.class in which base data set drops the AGE variable but still PROC COMPARE generates the above mentioned statement in its output by treating BASE data set as reference.

```
proc compare base=sashelp.class(drop=age) compare=sashelp.class;
run;
```

Even though the number of variables are not same still the statement “NOTE: No unequal values were found. All values compared are exactly equal.” is reported in the output of PROC COMPARE since it matches only the number of variables and observations from base dataset.

```
                          Data Set Summary

Dataset                 Created          Modified  NVar    NObs  Label

SASHELP.CLASS   24MAY11:13:52:29  24MAY11:13:52:29     4      19  Student Data
SASHELP.CLASS   24MAY11:13:52:29  24MAY11:13:52:29     5      19  Student Data


                          Variables Summary

                  Number of Variables in Common: 4.
                  Number of Variables in SASHELP.CLASS but not in SASHELP.CLASS: 1.


                         Observation Summary

                     Observation      Base  Compare

                     First Obs           1        1
                     Last  Obs          19       19

      Number of Observations in Common: 19.
      Total Number of Observations Read from SASHELP.CLASS: 19.
      Total Number of Observations Read from SASHELP.CLASS: 19.

      Number of Observations with Some Compared Variables Unequal: 0.
      Number of Observations with All Compared Variables Equal: 19.

      NOTE: No unequal values were found. All values compared are exactly equal.
```

Display 3: Proc Compare Misleading Output

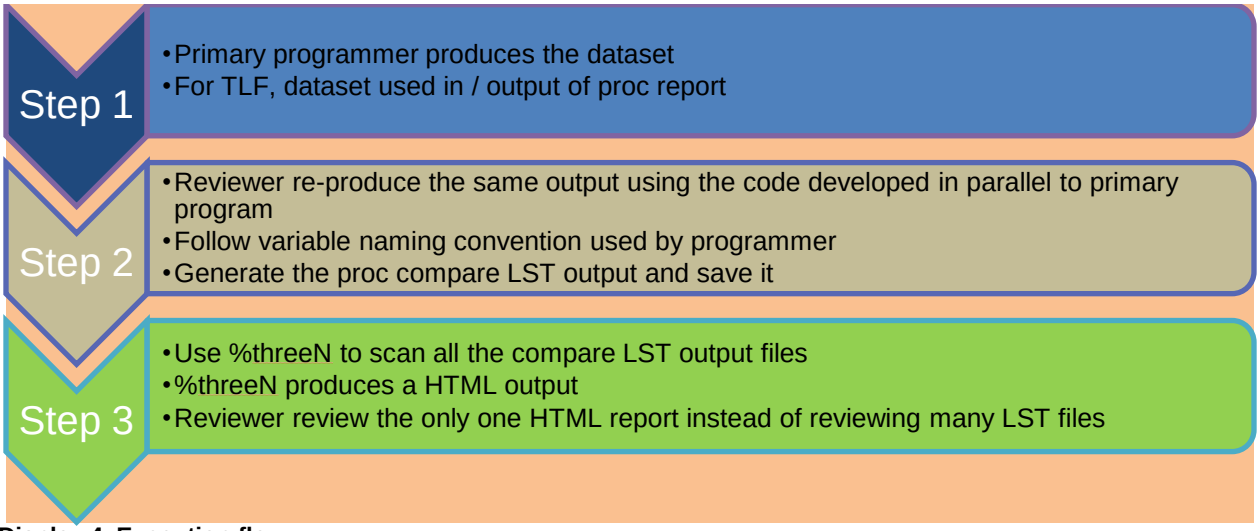
CHALLENGES IN VALIDATION

Each clinical study involves creating or reporting of numerous data sets, tables, listings and figures and their validations.

1. Manual checking of 100s of output or redoing the same while regenerating the same output is tedious, time consuming and error prone
2. PROC COMPARE can also be misleading the programmer with the above kind of outputs
3. The supervisor or team lead should await for the update from the programmers to know the status

PROPOSED SOLUTION

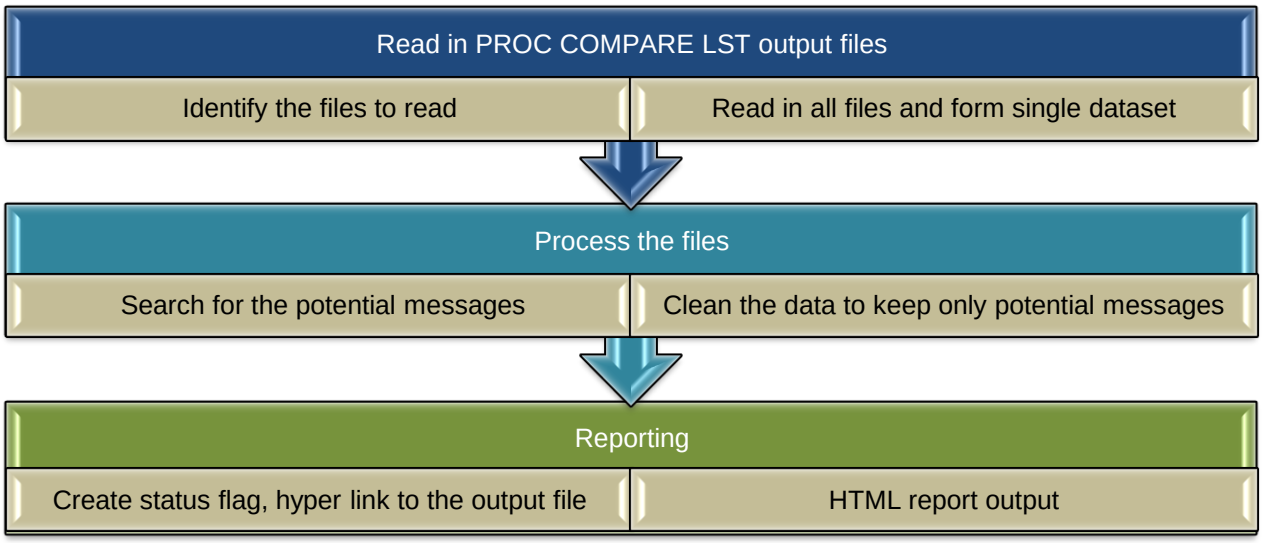
With **%threeN** macro, the above challenges are over come. The steps involved in the solutions are outlined as follows:



Display 4. Execution flow

%THREEN MACRO

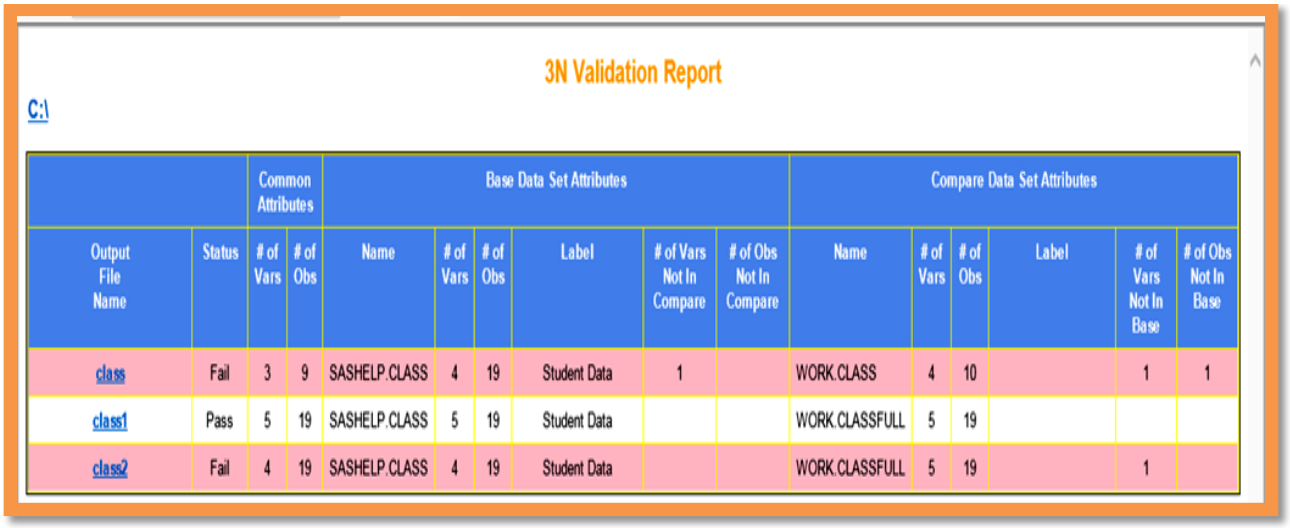
The functionality of %threeN macro is as follows



Display 5. %threeN Macro Process Flow

HTML OUTPUT

%threeN macro creates the following HTML file. Each output file is hyperlinked to its corresponding file and it helps to easily open the file from the 100s of file for the further review if it all needed.



3N Validation Report

C:\

		Common Attributes		Base Data Set Attributes						Compare Data Set Attributes					
Output File Name	Status	# of Vars	# of Obs	Name	# of Vars	# of Obs	Label	# of Vars Not In Compare	# of Obs Not In Compare	Name	# of Vars	# of Obs	Label	# of Vars Not In Base	# of Obs Not In Base
class	Fail	3	9	SASHELP.CLASS	4	19	Student Data	1		WORK.CLASS	4	10		1	1
class1	Pass	5	19	SASHELP.CLASS	5	19	Student Data			WORK.CLASSFULL	5	19			
class2	Fail	4	19	SASHELP.CLASS	4	19	Student Data			WORK.CLASSFULL	5	19		1	

Display 6. %threeN Output

The following measures of the base and compared data sets are reported.

- Status variable with Pass / Fail flag whenever the output file meets the following 3N criteria
  - ✓ **N**OTE: No unequal values were found. All values compared are exactly equal
  - ✓ **N**umber of observations in Base and Compared data sets are equal
  - ✓ **N**umbers of variables in Base and Compared data sets are equal

In addition to that it provides overall summary of both base and compares data sets.

CONCLUSION

The SAS macro %threeN efficiently serves the purpose of validating all the output files generated from PROC COMPARE in a short time and also expedite the process of validation with accuracy. It reduces up to 90% of time spend on the manual review of the output files. It can also be integrated with other programs like the program for checking the log files and making of any project status reports. It has also a limitation like the primary program dataset output and validation program output dataset should be of same form and it does the checks only on the data display and not the cosmetics which have to be handled by the programmer.

CONTACT INFORMATION

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

Amarnath Vijayarangan
Emmes Services Pvt Ltd, Bangalore, India
avijayarangan@emmes.com
amarnath7@gmail.com

APPENDIX:

%THREEN MACRO

```
%let path=C:\;

%macro threeN;
   filename pth "&path";

   *** Prepare list of output files to read;
   data lstList;
        length lstname  $ 500;
        dval=dopen('pth');
        if dval> 0 then do;
                 count=dnum(dval);
                 do i= 1 to count;
                          lstname=dread(dval,i);
                          if upcase(scan(lstname,2,'.'))='LST' then output;
                 end;
        end;
        keep lstname;
   run;

   filename pth clear;

   *** Read in all output files;
   data lst;
        length filepath $ 500;
        set lstlist;
        filepath=cats("&path.\",lstname);
        infile dummy filevar=filepath length=len end=done missover;
        do while (not done);
                 input record $varying2000. len;
                 filename=filepath;
                 record=strip(compbl(record));
                 if record^='' then output;
        end;
   run;

   proc sort data=lst;
        by filename;
   run;

   *** Ensure output file contains only one proc compare;
   proc freq data=lst noprint;
        tables filename /out=qc(drop=percent);
        where record='Data Set Summary';
```

```
run;

*** Delete filename from further process if more than one proc compare presents;
data lst;
     merge lst qc;
     by filename;
     if count=2 then delete;
run;

*** Create the necessary flags;
data lst;
     length BDataset BNvar BNObs BLabel CDataset CNvar CNObs CLabel
                CommonVars CommonObs BNotCVars CNotBVars BNotCObs CNotBObs $ 100;
     set lst;
     by filename;
     if record='The COMPARE Procedure' then id1=100;
             else id1+1;
     if record='Value Comparison Results for Variables' then id2=100;
             else id2+1;
     if record='Dataset Created Modified NVar NObs Label' then id3=100;
             else id3+1;
     retain BDataset BNvar BNObs BLabel CDataset CNvar CNObs
               CLabel CommonVars BNotCVars CNotBVars CommonObs BNotCObs CNotBObs;
     if first.filename then do;
             BDataset=''; BNvar='';BNObs=''; BLabel='';
             CDataset=''; CNvar=''; CNObs=''; CLabel='';
             CommonVars=''; BNotCVars=''; CNotBVars=''; CommonObs=''; BNotCObs='';
             CNotBObs='';
     end;

     if id3=101 then do;
             BDataset=strip(scan(record,1,''));
             BNvar=strip(scan(record,4,''));
             BNObs=strip(scan(record,5,''));
             BLabel=strip(substr(strip(record),(length(strip(record)))
     -index(strip(reverse(record)),strip(reverse(BNobs)))));
             if length(strip(BLabel))>length(strip(BNobs)) then
                     BLabel=substr(BLabel,1+length(strip(BNobs)));
                     else Blabel='';
     end;
     if id3=102 then do;
             CDataset=strip(scan(record,1,''));
             CNvar=strip(scan(record,4,''));
             CNObs=strip(scan(record,5,''));
             CLabel=strip(substr(strip(record),(length(strip(record)))
     -index(strip(reverse(record)),strip(reverse(CNobs)))));
             if length(strip(CLabel))>length(strip(CNobs)) then
                     CLabel=substr(CLabel,1+length(strip(CNobs)));
                     else Clabel='';
     end;
     if record=:'Number of Variables in Common' then
              CommonVars=strip(compress(scan(record,2,':'),,'kd'));
     if record=:catx('','Number of Variables in',BDataset, 'but not in',CDataset)
              then BNotCVars=strip(compress(scan(record,2,':'),,'kd'));
     if record=:catx('','Number of Variables in',CDataset, 'but not in',BDataset)
              then CNotBVars=strip(compress(scan(record,2,':'),,'kd'));

     if record=:'Number of Observations in Common' then
             CommonObs=strip(compress(scan(record,2,':'),,'kd'));
     if record=:catx('','Number of Observations in',BDataset, 'but not in',CDataset)
             then BNotCObs=strip(compress(scan(record,2,':'),,'kd'));
     if record=:catx('','Number of Observations in',CDataset, 'but not in',BDataset)
             then CNotBObs=strip(compress(scan(record,2,':'),,'kd'));
     if record=:catx('','Number of Variables in',BDataset, 'but not in',CDataset)
             then BNotCVars=strip(compress(scan(record,2,':'),,'kd'));
     if record=:catx('','Number of Variables in',CDataset, 'but not in',BDataset)
            then CNotBVars=strip(compress(scan(record,2,':'),,'kd'));
     if record=:'NOTE:' then Status='No unequal values were found';
     if Status='No unequal values were found' and (BNvar=CNvar) and (BNObs=CNObs)
             then Status='Pass';
             else Status='Fail';
     drop id:;
     if last.filename;
     lstname=scan(lstname,1,'.');
     lstname=cats('<a href=',cats("'",filename,"'"),'>',lstname,'</a>');
run;

filename temp url     'http://support.sas.com/rnd/base/ods/odsmarkup/tableeditor/tableeditor.tpl';
%include temp;

*** Create HTML report output;
```

```
   ods tagsets.tableeditor file="&path\3N Validation.html" style=styles.sasweb
        options( banner_color_odd="pink" GRIDLINE="YES" GRIDLINE_COLOR="yellow"
                 highlight_color='pink' frozen_headers="YES" frozen_rowheaders="YES");
   title1 h = 15pt c=Orange bold "3N Validation Report" ;
   title2 h = 12pt c=purple j=l bold "<a href=&path.>&path.</a>" ;
   proc report data=lst nowd split='*';
   column lstname status ('Common Attributes' CommonVars CommonObs)
         ('Base Data Set Attributes' BDataset BNvar BNObs BLabel BNotCVars  BNotCObs)
       ('Compare Data Set Attributes' CDataset CNvar CNObs CLabel CNotBVars CNotBObs);
   define lstname         / display  style(column)=
        {font_weight=bold just=center cellwidth=3.8 in} 'Output*File*Name' ;
   define status          / display  style(column)={just=center cellwidth=0.6 in} 'Status';
   define CommonVars      / display
        style(column)={just=center cellwidth=0.3 in} '# of Vars';
   define CommonObs       /display  style(column)={just=center cellwidth=0.3 in} '# of Obs';
   define BDataset /display  style(column)={just=center cellwidth=0.8 in} 'Name';
   define BNvar /display  style(column)={just=center cellwidth=0.3 in} '# of Vars';
   define BNObs /display  style(column)={just=center cellwidth=0.3 in} '# of Obs';
   define BLabel /display  style(column)={just=center cellwidth=2.8 in} 'Label';
   define BNotCVars  /display
        style(column)={just=center cellwidth=0.8 in} '# of Vars*Not In*Compare';
   define BNotCObs/display
        style(column)={just=center cellwidth=0.8 in} '# of Obs*Not In*Compare';
   define CDataset /display  style(column)={just=center cellwidth=0.8 in} 'Name';
   define CNvar /display  style(column)={just=center cellwidth=0.3 in} '# of Vars';
   define CNObs /display  style(column)={just=center cellwidth=0.3 in} '# of Obs';
   define CLabel /display  style(column)={just=center cellwidth=2.8 in} 'Label';
   define CNotBVars  /display
         style(column)={just=center cellwidth=0.9 in} '# of Vars*Not In*Base';
   define CNotBObs       /display
        style(column)={just=center cellwidth=0.9 in} '# of Obs*Not In*Base';
   run;
   ods tagsets.tableeditor close;
   ods html close;
%mend threeN;
%threeN;
```