PharmaSUG China 2024 - Paper AP-10022

Validation result checking and tracking by X command

Fangrun He, Sanofi

ABSTRACT

During the statistical programming of generating SDTM/ADaM datasets and Tables/Listings/Figures in clinical research, it always takes a lot of time for programmers to check the production and QC comparison results and make tracking files for datasets or TLFs after batch run for every milestone. To improve the efficacy and accuracy, various methods are raised by statistical programmers.

This paper aims to provide a method by calling X command in SAS to make tracking files and compare production and QC datasets/TLFs. The comparison results of batch run are often saved in RTF format or LOG format. We can use X command to specify a UNIX command to get the basic attributes such as file names, user id, modified timestamps of intended files and assign the “passing QC flag” based on comparison result files. In the end, we export all the results and save in a tracking EXCEL file.

INTRODUCTION

X Command in SAS enables you to enter UNIX commands without ending the SAS session. When you enter the X command, SAS starts a shell to execute the commands that you specify. The commands that you enter are processed differently, depending on whether you enter one command or more than one command. There are a lot of UNIX commands which help users gather information about their host operating system: listing files or directories to check attributes, processing text syntax to read text files line-by-line and use regular expressions to perform operations on specific parts of the file.

This paper will explain how to use X command to perform files attribute obtaining and searching QC comparison results. Firstly, we need to batch run the related SAS programs to obtain the needed datasets or TLFs in a specified directory and QC comparison results in RTF format or LOG format. And next we use X command to get the target columns (filename, timestamp, lead programmer id) of target datasets or TLFs, and then obtain the related QC columns (QC timestamp, QC programmer id) and check whether they passed QC or not. Finally make a tracking EXCEL file for them.

OVERVIEW OF THE PROGRAM

Step 1, define three macro variables for the directories of the data files, checking files and temporary files.

```
%let datadir = ROOT/XXX/COMPOUND/STUDY/ANALYSIS/ADS/DATA;
%let qcfidir = ROOT/XXX/COMPOUND/STUDY/ANALYSIS/QC/OUTPUT;
%let tempdir = ROOT/XXX/COMPOUND/STUDY/ANALYSIS/MISC;
```

Step 2, find the file details of the datasets or TLFs and save them into a temp file named temp1.txt, and then import into SAS datasets.

```
x "stat -c %n@%z@%U /path/ &datadir./*.sas7bdat > &tempdir./temp1.txt";
proc import out=my_data
    datafile="&tempdir./temp1.txt"
    dbms=csv
    replace;
    getnames=NO;
   guessingrows = max;
run;
```

```
data readfile;
    set my_data;
    filenam = prxchange("s/^\/home.+DATA\/(.+)\.sas7bdat/$1/",-
1,scan(var1, 1, '@'));
    timestamp = scan(var1, 2, '@');
    id = scan(var1, 3, '@');
    drop var1;
run;
```

Step 3, find the file details of QC files and save them into a temp file named temp2.txt, and then import into SAS datasets.

```
x "stat -c %n@%z@%U /path/ &qcfidir./*.rtf > &tempdir./temp2.txt";

proc import out=qc_data
    datafile="&tempdir./temp2.txt"
    dbms=csv
    replace;
    getnames=NO;
    guessingrows = max;
run;

data qcfile;
    set qc_data;
    filenam = prxchange("s/^\/home.+OUTPUT\/qc\_(.+)\.rtf/$1/",-
1,scan(var1, 1, '@'));
    qc_timestamp = scan(var1, 2, '@');
    qc_id = scan(var1, 3, '@');
    drop var1;
run;
```

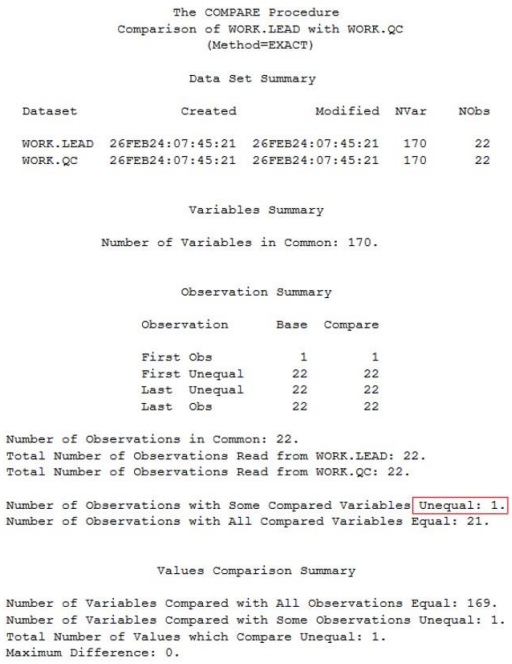
Step 4, find the comparison results which do not pass QC, and save them into a temp file named temp3.txt, and then import into SAS datasets.

```
x "grep -rilE 'Unequal: [1-9]' /path/ &qcfidir./*.rtf
   > &tempdir./temp3.txt";

x "grep -ril 'but not in' /path/ &qcfidir./*.rtf
   >> &tempdir./temp3.txt";

x "grep -ril 'in Common: 0' /path/ &qcfidir./*.rtf
   >> &tempdir./temp3.txt";

x "grep -ril 'Conflicting Types' /path/ &qcfidir./*.rtf
   >> &tempdir./temp3.txt";

proc import out=not_pass
    datafile="&tempdir./temp3.txt"
    dbms=dlm
    replace;
    getnames=NO;
    guessingrows = max;
run;
```

```
data notpfile;
    set not_pass;
    filename = prxchange("s/^\/.+\/qc\_(.+)\.rtf/$1/",-1,var1);
    id = prxchange("s/^\/home\/(.+)\/wise.+$/$1/",-1,var1);
run;
```

Here we have 4 scenarios of comparison results which do not pass QC in X command.

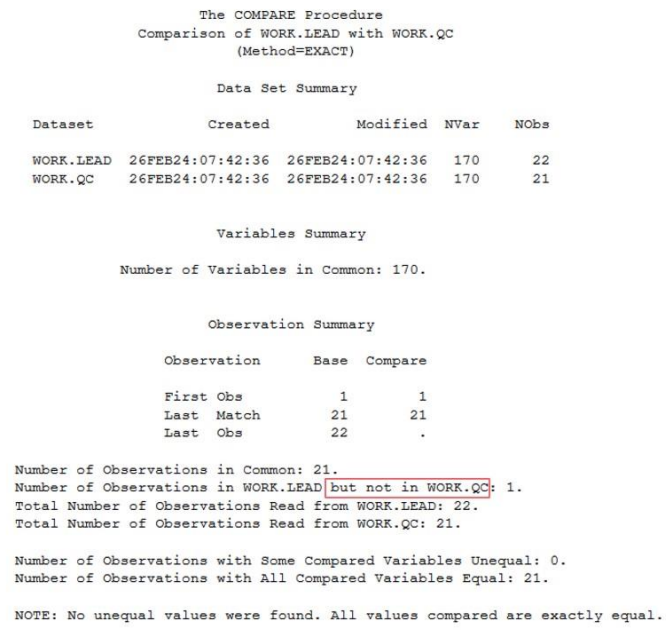
Scenario 1: there are unequal records in both lead and QC datasets, which can be found in Figure 1. Programmers need to check the unequaled variables.



```
                    The COMPARE Procedure
                Comparison of WORK.LEAD with WORK.QC
                          (Method=EXACT)

                        Data Set Summary

Dataset             Created          Modified  NVar    NObs

WORK.LEAD  26FEB24:07:45:21  26FEB24:07:45:21   170      22
WORK.QC    26FEB24:07:45:21  26FEB24:07:45:21   170      22


                        Variables Summary

            Number of Variables in Common: 170.


                        Observation Summary

                  Observation      Base  Compare

                  First Obs           1        1
                  First Unequal      22       22
                  Last  Unequal      22       22
                  Last  Obs          22       22

Number of Observations in Common: 22.
Total Number of Observations Read from WORK.LEAD: 22.
Total Number of Observations Read from WORK.QC: 22.

Number of Observations with Some Compared Variables Unequal: 1.
Number of Observations with All Compared Variables Equal: 21.


                    Values Comparison Summary

Number of Variables Compared with All Observations Equal: 169.
Number of Variables Compared with Some Observations Unequal: 1.
Total Number of Values which Compare Unequal: 1.
Maximum Difference: 0.
```

Figure 1

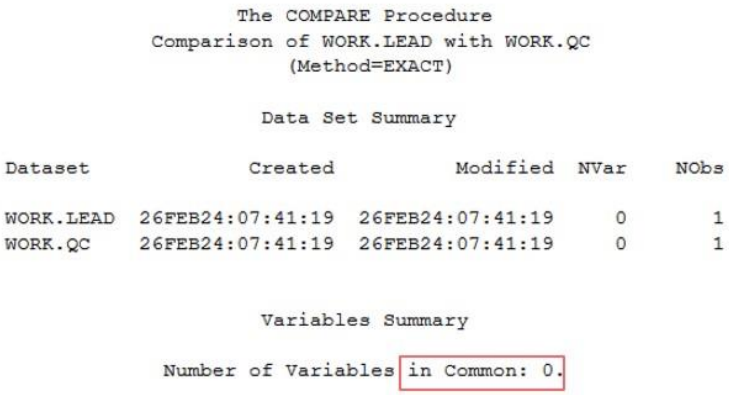
Scenario 2: there are missing records in any of the datasets, which can be found in Figure 2. Programmers need to check why the records are missing.



```
                The COMPARE Procedure
            Comparison of WORK.LEAD with WORK.QC
                      (Method=EXACT)

                    Data Set Summary

 Dataset            Created          Modified  NVar    NObs

 WORK.LEAD  26FEB24:07:42:36  26FEB24:07:42:36   170      22
 WORK.QC    26FEB24:07:42:36  26FEB24:07:42:36   170      21


                    Variables Summary

          Number of Variables in Common: 170.


                    Observation Summary

               Observation      Base  Compare

               First Obs           1        1
               Last  Match        21       21
               Last  Obs          22        .

Number of Observations in Common: 21.
Number of Observations in WORK.LEAD but not in WORK.QC: 1.
Total Number of Observations Read from WORK.LEAD: 22.
Total Number of Observations Read from WORK.QC: 21.

Number of Observations with Some Compared Variables Unequal: 0.
Number of Observations with All Compared Variables Equal: 21.

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

Figure 2

Scenario 3: both lead and QC datasets are empty, which can be found in Figure 3. Programmers need to check whether they should be empty or not.



```
                The COMPARE Procedure
            Comparison of WORK.LEAD with WORK.QC
                      (Method=EXACT)

                    Data Set Summary

 Dataset            Created          Modified  NVar    NObs

 WORK.LEAD  26FEB24:07:41:19  26FEB24:07:41:19     0       1
 WORK.QC    26FEB24:07:41:19  26FEB24:07:41:19     0       1


                    Variables Summary

          Number of Variables in Common: 0.
```

Figure 3

Scenario 4: conflicting types exist in lead and QC datasets, which can be found in Figure 4. Programmers need to update the conflicting variables.

```
                    The COMPARE Procedure
                Comparison of WORK.LEAD with WORK.QC
                            (Method=EXACT)

                         Data Set Summary

  Dataset             Created          Modified  NVar    NObs

  WORK.LEAD  26FEB24:07:24:57  26FEB24:07:24:57   170      22
  WORK.QC    26FEB24:07:26:06  26FEB24:07:26:06   170      22


                         Variables Summary

          Number of Variables in Common: 170.
          Number of Variables with Conflicting Types: 1.


        Listing of Common Variables with Conflicting Types

            Variable  Dataset    Type  Length  Label

            AGE       WORK.LEAD  Num        8  Age
                      WORK.QC    Char      12


                       Observation Summary

                  Observation      Base  Compare

                  First Obs           1        1
                  Last  Obs          22       22

Number of Observations in Common: 22.
Total Number of Observations Read from WORK.LEAD: 22.
Total Number of Observations Read from WORK.QC: 22.

Number of Observations with Some Compared Variables Unequal: 0.
Number of Observations with All Compared Variables Equal: 22.

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

Figure 4

Step 5, merge the dataset information with QC status and results, and then export as tracking.xlsx file.

```
proc sql;
    create table tracking as
    select a.*, b.qc_timestamp, b.qc_id, case when c.filename is not
null then "N" else "Y" end as pass_flag
     from readfile as a
     left join qcfile as b on a.filenam=b.filenam
     left join notpfile as c on lowcase(a.filenam) = c.filename
     ;
quit;

proc export data=tracking
    outfile="&tempdir./tracking.xlsx"
    dbms=xlsx
    replace;
    sheet="My Sheet";
run;
```

The tracking.xlsx file is shown in below Figure 5.

	A	B	C	D	E	F
1	filenam	timestamp	id	qc_timestamp	qc_id	pass_flag
2	adae	2023-12-10 00:40:40.000000000 +0100	id1	2023-12-10 00:40:39.000000000 +0100	id2	Y
3	adcc	2023-12-10 00:40:40.000000000 +0100	id1	2023-12-10 00:40:39.000000000 +0100	id2	Y
4	adcm	2023-12-10 00:40:41.000000000 +0100	id1	2023-12-10 00:40:39.000000000 +0100	id2	Y
5	adcv	2023-12-10 00:40:40.000000000 +0100	id1	2023-12-10 00:40:39.000000000 +0100	id2	Y
6	addv	2023-12-10 00:40:40.000000000 +0100	id1	2023-12-10 00:40:39.000000000 +0100	id2	Y
7	adeg	2023-12-10 00:40:40.000000000 +0100	id1	2023-12-10 00:40:39.000000000 +0100	id2	Y
8	adex	2023-12-10 00:40:40.000000000 +0100	id1	2023-12-10 00:40:39.000000000 +0100	id2	Y
9	adft	2023-12-10 00:40:41.000000000 +0100	id1	2023-12-10 00:40:39.000000000 +0100	id2	Y
10	adis	2023-12-10 00:40:40.000000000 +0100	id1	2023-12-10 00:40:39.000000000 +0100	id2	Y
11	adlb	2023-12-10 00:40:41.000000000 +0100	id1	2023-12-10 00:40:39.000000000 +0100	id2	Y
12	adlbus	2023-12-10 00:40:41.000000000 +0100	id1	2023-12-10 00:40:39.000000000 +0100	id2	Y
13	adoe	2023-12-10 00:40:40.000000000 +0100	id1	2023-12-10 00:40:39.000000000 +0100	id2	Y
14	adpc	2023-12-10 00:40:40.000000000 +0100	id1	2023-12-10 00:40:39.000000000 +0100	id2	Y
15	adpe	2023-12-10 00:40:40.000000000 +0100	id1	2023-12-10 00:40:39.000000000 +0100	id2	Y
16	adpf	2023-12-10 00:40:40.000000000 +0100	id1	2023-12-10 00:40:39.000000000 +0100	id2	Y
17	adpp	2023-12-10 00:40:40.000000000 +0100	id1	2023-12-10 00:40:39.000000000 +0100	id2	Y
18	adpr	2023-12-10 00:40:40.000000000 +0100	id1	2023-12-10 00:40:39.000000000 +0100	id2	Y
19	adqs	2023-12-10 00:40:40.000000000 +0100	id1	2023-12-10 00:40:39.000000000 +0100	id2	Y
20	adre	2023-12-10 00:40:40.000000000 +0100	id1	2023-12-10 00:40:39.000000000 +0100	id2	Y

Figure 5

CONCLUSION

This program tool helps SAS programmers deliver high qualified validation results efficiently and bring conveniences to programmers in daily work.

REFERENCES

Li Qianqian, Zou Qing, Liu Zhimin. Automation of validation result checking by Python, Available from:

https://pharmasug.com.cn/resources/2023/2023-09/Pharmasug-China-2023-PO114.pdf

CONTACT INFORMATION

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

Fangrun He
Sanofi
Fangrun.He@sanofi.com
39F, Chengdu Yintai in99 Tower 3, Chengdu, Sichuan, China