PharmaSUG 2015 - Paper QT36

Mlibcompare – keep an eye on the changes of your data

Wen Shi, Accenture, Berwyn, PA

ABSTRACT

When developing the regular submission package, we usually follow the order of generating SDTM first, then ADaM, then the TLF outputs. In real practice, we may keep receiving many cuts of new data and still need to guarantee the function of our programs. Macro Mlibcompare will help you quickly identify the structural changes of your data sets in your library. It combines the results from several SAS procedures, analyzes them and generates an excel summary report with the Microsoft Excel DDE interface.

INTRODUCTION

To meet FDA's requirements, clinical data submitted needs to be presented in compliance with the CDSIC standards. Raw data collected in the clinical study will be cleaned, coded and converted into STDM data sets. From the SDTM data sets, we build the ADaM data sets and further create the analysis outputs such as tables, listings and figures. Most of the SAS programs are functioning as "Machines" processing the input data during each of these steps.

We write various kinds of SAS programs every day. A fact is that after a program is finished, it is seldom run just once. Many times of re-run will happen most likely due to the change of the input data. There are several reasons causing a change, for example, at the time one project was started, the database has not been locked; or, the programs were created aiming generating previews repeatedly at different time points; or, some data issues were found and corrected in the input data thus a re-run was requested to update the outputs.

Whenever a change happens, your program might expire. Sometimes the program can still be run without showing any warning or error; however, incorrect result might have been generated without your awareness.

Let's take a look at some sample issues happened during the re-run with changed data.

ISSUE #1

```
154  data LB1;
155       merge LB SUPPLB;
ERROR: File WORK.SUPPLB.DATA does not exist.
156       by STUDYID USUBJID LBSEQ;
157  run;

NOTE: The SAS System stopped processing this step because of errors.
WARNING: The data set WORK.LB1 may be incomplete.  When this step was stopped there were 0 observations and 30 variables.
NOTE: DATA statement used (Total process time):
      real time           0.01 seconds
      cpu time            0.03 seconds
```

Figure 1. Sample Error from SAS log Caused by the Deletion of A Data Set

This issue happened when we were creating the ADLB data set (ADaM) by merging LB domain and SUPPLB domain (SDTM) together. The SUPPLB data set was later decided to be removed from the newest cut of data, causing a problem with the re-run. Removing or renaming of data sets, variables or values can cause similar situations like this.

ISSUE #2

Age (years)	n	0	0	0	0	0
	Mean	0	0	0	0	0
	SD	0	0	0	0	0
	Median	0	0	0	0	0
	Min.	0	0	0	0	0
	Max.	0	0	0	0	0

Figure 2. Sample Part from a Demographic Table with Implausible Numbers

In this case, there was no explicit error shown in the SAS log. However, apparently an incorrect result has been generated in the table. This is because in the program a condition – where AVALCAT1 = “North America” – has been used. The value “North America” in the condition matched one of the values in AVALCAT1 in the previous cut of data. However, “North America” was changed to “NORTH AMERICA” in AVALCAT1 later in the new cut of data, consequently causing a problem when re-running the table program. Similar cases would be the changing of the visit number, sub groups of population or population flags, etc.

ISSUE #3

Reasons for Delay [2]				
n	671	369	707	416
HAEMATOLOGIC ADVERSE EVENT	212 (32%)	112 (30%)	276 (39%)	114 (27%)
CARDIAC ADVERSE EVENT	1 (<1%)	4 (1%)	1 (<1%)	2 (<1%)
ADVERSE EVENT OTHER THAN HAEMATOLOGIC OR	299 (45%)	107 (29%)	280 (40%)	139 (33%)
DOSING ERROR	0	0	0	0
SUBJECT NON-COMPLIANCE	79 (12%)	73 (20%)	78 (11%)	81 (19%)
ADMINISTRATIVE REASONS	63 (9%)	68 (18%)	59 (8%)	72 (17%)
OTHER	11 (2%)	5 (1%)	6 (<1%)	5 (1%)
UNKNOWN	0	0	0	0

Figure 3. Sample Part from a Table with an Unexpected Truncation

In the highlighted row of the screenshot, a truncation happened with the text in the column called “Reasons for Delay”. The length of the column was set to accommodate the values other than “ADVERSE EVENT OTHER THAN HAEMATOLOGIC OR CARDIAC” because in the previous cut of data there’s no such value appeared. However, this value showed up in the new cut of data and got truncated in the updated table. So a necessary adjustment has to be made in this summary table program.

MLIBCOMPARE

As what’s shown above, in general, all the changes happened in the input data can be grouped in to three levels: data set level, variable level and value level. In order to reflect these changes, three different SAS procedures are used recursively in the Mlibcompare macro to get the meta data of each level for comparison. Then these meta data will be merged together at each level to get the differences. Finally, the differences will be filtered out , and the result will be populated into an Excel template. A Summary report will be generated and saved on the first sheet. For each individual data set where any changes happened, a separate sheet will be created, containing the information from the comparison of each level. The entire output procedure will be carried out by using the Excel DDE.

The details of the three SAS procedures are as following:

DATASET LEVEL PROCEDURE

```
ods output "Library Members" = libmeta_01;
proc datasets library = &libname details;
   quit;
ods output close;
```

Figure 4. Sample Code for Data set Level Procedure

This procedure will be used to check the changes such as added data sets, removed data sets, changes in labels, changes in the total number of variables, changes in the total number of observations.

VARIABLE LEVEL PROCEDURE

```
proc contents data = &dsname out = dsmeta_01 noprint;
run;
```

Figure 5. Sample Code for Variable Level Procedure

This procedure will be used to check the changes such as added variable, removed variable, changes in variable types, lengths, labels, formats, etc.

VALUE LEVEL PROCEDURE

```
proc freq data = &dsname noprint;
    table &varname/out=varmeta_01;
run;
```

Figure 6. Sample Code for Value Level Procedure

This procedure will be used to check the changes such as new values, removed old values, new missing value, missing value fixed, etc.

SAMPLE OUTPUTS:

This section gives a sample of the report generated by Mlibcompare.

1		
2	**Base lib:**	F:\Data\SDTM\Archive\2014-07-16
3	**Compare lib:**	F:\Data\SDTM\Archive\2014-08-19
4	**Avoid datasets:**	
5	**Avoid variables:**	
6	**Start date:**	Tuesday, 19 August 2014
7	**Start time:**	12:26:42
8	**End date:**	Tuesday, 19 August 2014
9	**End time:**	12:29:17
10	**Time elapsed:**	155 s
11		
12		
13	**Summary:**	
14	# of Datasets Compared :	62
15	# of Datasets with Difference(s) :	18 (29%)
16	# of Datasets without Difference(s) :	44 (71%)
17	# of Datasets with Major Difference(s) :	18 (29%)
18	# of Datasets with Minor Difference(s) :	11 (18%)

20	**LEVEL - 0**	**Differences Found in DATASET Level Metadata**		
21	**Significance**	**Details**	**List of Datasets with difference(s)**	**Total (datasets)**
22	Major	DATASET ADDED		0
23	Major	DATASET REMOVED	PE, SUPPFAPR, SUPPMH, SUPPPC, SUPPPE	5
24	Major	NUMBERS OF VARIABLES INCREASED	PC, RS, TR, TU	4
25	Major	NUMBERS OF VARIABLES REDUCED		0
26	Major	LABEL CHANGED		0
27	Major	OBSERVATIONS ADDED	PC, SUPPDA, SV, XG	4
28	Major	OBSERVATIONS REMOVED	EX, SUPPTR	2
29				
30				
31				
32	**LEVEL - 1**	**Differences Found in VARIABLE Attributes**		
33	**Significance**	**Details**	**List of Datasets with difference(s)**	**Total (datasets)**
34	Major	VARIABLE ADDED	PC, RS, TR, TU	4
35	Major	VARIABLE REMOVED	PC	1
36	Major	Type		0
37	Major	Length	DM, EX, IE, PC, RS, SUPPEG, TR	7
38	Major	Label		0
39	Major	Format		0
40	Major	Format Length		0
41	Major	Number of Decimals in Format		0
42	Minor	Informat		0
43	Minor	Informat Length		0
44	Minor	Number of Decimals in Informat		0
45	Minor	Status - No Duplicate Keys		0
46	Minor	Status - No Duplicate Records		0
47	Minor	Sorting Status		0
48	Minor	Position in the Sort by Clause		0

summary | DM | EX | IE | LB | PC | PE | RS | SUPPDA | SUPPEG | SUPPFAPR | SUPPMH | SUPPPC | SUPPI

Figure 7. Sample Part of the Report from Summary Sheet

Leve	Signifi	Variable Name	Description	Details	Value	Filter
0	Major		Numbers Of Variables Increased	VARS: BASE=20 COMP=21		
1	Major	EPOCH	Variable Added	TYPE: BASE= COMP=Char		
1	Major	RSREASND	Difference Found : Length Of Variable	LENGTH: BASE=40 COMP=36		
1	Major	RSSTRESC	Difference Found : Length Of Variable	LENGTH: BASE=19 COMP=13		
2	Major	RSREASND	Value Removed	COUNTS: BASE=205 COMP=.	SUBJECT DOES NOT HAVE NON-TARG	Other
2	Major	RSSTRESC	Value Removed	COUNTS: BASE=160 COMP=.	PARTIAL RESPONSE	–STRES–
2	Major	RSSTRESC	Value Removed	COUNTS: BASE=7 COMP=.	NOT EVALUABLE	–STRES–
2	Major	RSSTRESC	Value Removed	COUNTS: BASE=119 COMP=.	STABLE DISEASE	–STRES–
2	Major	RSSTRESC	Value Removed	COUNTS: BASE=61 COMP=.	PROGRESSIVE DISEASE	–STRES–
2	Minor	RSREASND	Counts Increased	COUNTS: BASE=166 COMP=528	NOT APPLICABLE	Other
2	Minor	RSREASND	Counts Decreased	COUNTS: BASE=280 COMP=123	SUBJECT DOES NOT HAVE TARGET LE	Other
2	Minor	RSSTRESC	Counts Increased	COUNTS: BASE=632 COMP=751	SD	–STRES–
2	Minor	RSSTRESC	Counts Increased	COUNTS: BASE=781 COMP=941	PR	–STRES–
2	Minor	RSSTRESC	Counts Increased	COUNTS: BASE=384 COMP=445	PD	–STRES–
2	Minor	RSSTRESC	Counts Increased	COUNTS: BASE=95 COMP=102	NE	–STRES–

Figure 8. Sample Part of the Report from Individual Data set Sheet

MLIBCOMPARE VS. PROC COMPARE

- PROC COMPARE compares one single data set against another one, whereas Mlibcompare compares an entire library against another data library.
- PROC COMPARE procedure compares the data sets line by line, value by value whereas Mlibcompare compares the meta-data regardless of the alignment.
- PROC COMPARE procedure does not provide an insight into the change whereas Mlibcompare provides an overview of all the changes.

INPUT PARAMETERS PASSED INTO THE MACRO

- Base library path (basepath=)
- Compare library path (comparepath=)
- Excel template path (templatepath=)
- List of data sets to avoid (avoiddslist=)
- List of variables to avoid (avoidvarlist=)
- Output path and file name (outfile=)

CONCLUSION

Mlibcompare can provide us a quick and overall view of changes happened between two cuts of data, helps predicting the possible changes needed to our programs.

However, there are some changes which cannot be detected by Mlibcompare.

For example:

Old data

USUBJID	CMSEQ	CMDECOD
001-0001	1	Drug A
001-0001	2	Drug B

New Data

USUBJID	CMSEQ	CMDECOD
001-0001	1	Drug B
001-0001	2	Drug A

In this case, the sorting status of the data set didn't change; the length of the variable didn't change; and the counts of the value didn't change as well. The only change is that the values in CMDECOD are switched. Mlibcompare can't detect such kind of changes. Fortunately, this kind of changes won't cause any error and might be something expected. Associating the changes with the combination of some key variables would be one of the improvements of this macro which we are going to cover in the future.

CONTACT INFORMATION

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

Name: Wen (Steven) Shi
Enterprise: Accenture LLP
Address: 1160 W. Swedesford Rd. Bldg. One, Berwyn PA
Work Phone: (610) 407-7535
E-mail: wen.s.shi@accenture.com

APPENDIX A

```
%macro mlibcompare(basepath=, comparepath=, avoiddslist=, avoidvarlist=,
templatepath=, outfile=);

*** get execution path of the program ***;
%macro setexecpath();
    %global execpath;
    %let execpath="";
    %let execpath=%sysfunc(getoption(SYSIN));
    %if %length(&execpath)=0
           %then %let execpath=%sysget(SAS_EXECFILEPATH);
%mend setexecpath;

%setexecpath();
%put &execpath;

*** set the program name and path in order to write the log into log file in
interactive mode ***;
%macro setSASAUTOVAR();
    %global SAS_EXECFILENAME;
    %global SAS_EXECFILEPATH;

    %let SAS_EXECFILENAME=%scan(&execpath, -1, "\");
    %put &SAS_EXECFILENAME;

    %let SAS_EXECFILEPATH=%sysfunc(tranwrd(&execpath, &SAS_EXECFILENAME,));
    %put &SAS_EXECFILEPATH;

%mend setSASAUTOVAR;

%setSASAUTOVAR();

%if &outfile eq %then %do;
    %let outfile=&SAS_EXECFILEPATH.result.xls;
    %put &outfile;
%end;

*** set log file name ***;
%let logn = %substr(&SAS_EXECFILENAME,1, %sysfunc(find(&SAS_EXECFILENAME,.sas,i))-
1);
%put &logn;

proc printto log="&logn..log" new;
run;


*** timer ****;
data timer;
    dt = datetime();
    format dt datetime.;
run;


*** get numbers of obs in a dataset ***;
%macro mnobs(inds=, nobsvar=);
    %global &nobsvar;

    %let dsid = %sysfunc(open(&inds));
    %let &nobsvar = %sysfunc(attrn(&dsid, nobs));
    %let rc = %sysfunc(close(&dsid));
```

```
%mend mnobs;

*** check a dataset is empty or not ***;
%macro mnodata(inds=, flag=);
    %global &flag;

    %let dsid = %sysfunc(open(&inds));
    %let nobs = %sysfunc(attrn(&dsid, nobs));
    %let rc = %sysfunc(close(&dsid));

    %if &nobs eq 0 %then %do;
           %let &flag = 'Y';
    %end;
    %else %do;
           %let &flag = 'N';
    %end;
%mend mnodata;

*** create type filter by checking the characteristics of variable name ***;
%macro dofilter(chkvar=);
    length filter $40;
%mend dofilter;


%macro libmeta(libn=, libpath=, avoidds=, avoidvar=);

    libname &libn &libpath ;

    ******************************************************;
    ***     Library-scale : dataset metadata reading    ***;
    ***     output : result_01                          ***;
    ******************************************************;

    ods output "Library Members" = result_01;
    proc datasets library = &libn details;
       quit;
    ods output close;

    data result_01;
           set result_01;
           %if &avoidds ne %then %do;
           where name not in &avoidds;
           %end;
    run;

    ******************************************************;
    ***     dataset-scale : variable metadata reading   ***;
    ***     output : result_02                          ***;
    ******************************************************;

    data result_02;
        stop;
    run;

    %macro pullout(inlib=);

        %mnobs(inds=result_01,nobsvar=nds);

        %do i = 1 %to &nds;

        data _null_;
            set result_01;
```

```
            if _n_ = &i then call symput('nam', name);
        run;

        proc contents data = &inlib..&nam out = meta_&inlib._&nam(drop =
            memlabel typemem engine crdate modate delobs idxusage memtype idxcount
flags reuse
            collate pointobs genmax gennum gennext NPOS) noprint;
        run;

        data result_02;
            set result_02 meta_&inlib._&nam;
        run;

           proc datasets nolist;
                 delete meta_&inlib._&nam;
           quit;
           run;

           data val_&inlib._&nam;
           stop;
           run;

        %end;
    %mend pullout;

    %pullout(inlib=&libn.);

    ******************************************************;
    ***     dataset-scale : value frequency counting    ***;
    ***     output : result_03                          ***;
    ***     all values : var_pool                       ***;
    ***          missing values : var_mis                                  ***;
    ******************************************************;
    data result_03;
        stop;
    run;

    %macro checkallvalues(inlib=, avoidds=, avoidvar=);

        %mnobs(inds=result_02,nobsvar=allvars);

        %do j = 1 %to &allvars;
            data _null_;
                set result_02;
                if _n_ eq &j. then do;
                    call symput('dsname',strip(memname));
                    call symput('var', strip(name));
                    call symput('varid', strip(put(varnum, 4.0)));
                    call symput('vartype', strip(put(type, 1.0)));
                    call symput('varfmt', strip(format));
                    call symput('varfmtl', strip(put(formatl, best.)));
                    call symput('varfmtd', strip(put(formatd, best.)));
                end;
            run;

            proc freq data = &inlib..&dsname noprint;
                table &var/out = varval;
            run;

            data temp(keep = libname dsname varname varid vartype varfmt varfmtl
varfmtd varval count)
                 varmis(keep = libname dsname varname count rename = (count =
nmiss));
```

```
                set varval;
                length libname dsname varname $100 varid $4 vartype $5 varfmt
varfmtl varfmtd $8 varval $400;
                libname = upcase(%unquote(%str(%')&inlib%str(%')));
                dsname = upcase(%unquote(%str(%')&dsname%str(%')));
                varname = upcase(%unquote(%str(%')&var%str(%')));
                varid = upcase(%unquote(%str(%')&varid%str(%')));

                %if &vartype eq 1 %then %do;
                    vartype = 'Num';
                %end;
                %else %if &vartype eq 2 %then %do;
                    vartype = 'Char';
                %end;

                varfmt = upcase(%unquote(%str(%')&varfmt%str(%')));
                varfmtl = upcase(%unquote(%str(%')&varfmtl%str(%')));
                varfmtd = upcase(%unquote(%str(%')&varfmtd%str(%')));

                %if &varfmt ne %then %do;
                    %put format=&varfmt lenth=&varfmtl decimal=&varfmtd;
                    varval = strip(put(&var, &varfmt.&varfmtl..&varfmtd));
                %end;
                %else %do;
                    %if &vartype eq 2 %then %do;
                        varval = strip(&var);
                    %end;
                    %else %do;
                        varval = strip(put(&var, best.));
                    %end;
                %end;
                        %if &avoidvar ne %then %do;
                if varname not in &avoidvar then output temp;
                        %end;
                        %else %do;
                        output temp;
                        %end;
                if varval in ('', '.') then output varmis;
            run;

          data val_&inlib._&dsname;
              retain libname dsname varname varid vartype varfmt varfmtl varfmtd
varval count;
              set val_&inlib._&dsname temp;
          run;

            data result_03;
                retain libname dsname varname nmiss;
                set result_03 varmis;
            run;

        %end;

    %mend checkallvalues;

    %checkallvalues(inlib=&libn, avoidds=&avoidds, avoidvar=&avoidvar);

%mend libmeta;

%libmeta(libn=base, libpath=&basepath, avoidds=&avoiddslist,
avoidvar=&avoidvarlist);

************************************************************;
```

```
***     COMPARE starts here.                                                ***;
***     rename first libmeta call result to base        ***;
***     rename second libmeta call result to comare     ***;
*************************************************************;

proc datasets library = work;
    change result_01=base_01 result_02=base_02 result_03=base_03;
    quit;
run;

%libmeta(libn=compare, libpath=&comparepath, avoidds=&avoiddslist,
avoidvar=&avoidvarlist);

proc datasets library = work;
    change result_01=compare_01 result_02=compare_02 result_03=compare_03;
    quit;
run;

data time_temp;
    set timer;
    sec = datetime() - dt;
run;

*********************************************;
*** level 1 : compare dataset metadata             ***;
*** output : diff_01                                    ***;
*********************************************;
proc sort data = base_01;
    by name;
run;

proc transpose data = base_01 out = b(rename = (_NAME_ = attr col1 = base) drop =
_LABEL_);
    by name;
    var num memtype obs vars label filesize lastmodified;
run;

proc sort data = compare_01;
    by name;
run;

proc transpose data = compare_01 out = c(rename = (_NAME_ = attr col1 = compare)
drop = _LABEL_);
    by name;
    var num memtype obs vars label filesize lastmodified;
run;

proc sort data = b;
    by name attr;
run;

proc sort data = c;
    by name attr;
run;

data process_01;
    length name attr $100;
    merge b(in = a) c(in = b);
    by name attr;
    base = strip(base);
    compare = strip(compare);
    length reason $400;
    if cmiss(base, compare) = 1 then do;
```

```
            if compare = '' then do;
                  prim_diff = 1; ord = 0; reason = 'DATASET REMOVED';
            end;
            else if base = '' then do;
                  prim_diff = 2; ord = 0; reason = 'DATASET ADDED';
            end;
    end;
    else if cmiss(base, compare) = 0 and base ne compare then do;
            if attr = 'MemType' then do; prim_diff = 3; ord = 0; reason = 'Difference
found : Member Type'; end;
            if attr = 'Obs' and (input(base, best.) > input(compare, best.))   then
do; prim_diff = 4.1; ord = 0; reason = 'OBSERVATIONS REMOVED'; end;
            if attr = 'Obs' and (input(base, best.) < input(compare, best.))   then
do; prim_diff = 4.2; ord = 0; reason = 'OBSERVATIONS ADDED'; end;
            if attr = 'Vars' and (input(base, best.) > input(compare, best.))  then
do; prim_diff = 5.1; ord = 0; reason = 'Numbers of Variables Reduced'; end;
            if attr = 'Vars' and (input(base, best.) < input(compare, best.))  then
do; prim_diff = 5.2; ord = 0; reason = 'Numbers of Variables Increased'; end;
            if attr = 'Label'   then do; prim_diff = 6; ord = 0; reason = 'Difference
found : Dataset Label'; end;

            if attr = 'FileSize'              then do; prim_diff = 7; ord = 0; reason
= 'Difference found : File Size of Dataset'; end;
            if attr = 'LastModified'   then do; prim_diff = 8; ord = 0; reason =
'Difference found : Last Modified Date/time of Dataset'; end;
    end;
run;

proc sort data = process_01 out = diff_01;
    by name prim_diff;
    where prim_diff >. ;
run;

data diff_01(drop = diffcnt);
    set diff_01;
    by name;
    retain diffcnt 0;
    if first.name then diffcnt = 1;
    else diffcnt = diffcnt + 1;

    if prim_diff = 3 then delete;
    if prim_diff = 7 then delete;
    if prim_diff = 8 then delete;
    if prim_diff in (1, 2) and attr ^= 'Obs' then delete;

    length cat $10;
    if prim_diff in (7, 8) then cat = 'Minor';
    else cat = 'Major';
run;

%mnodata(inds=diff_01, flag=nodiff_01);

data diff_01;
    if &nodiff_01 eq 'N' then do;
            set diff_01;
            reason = propcase(reason);
            length desc $400;
            desc = strip(upcase(attr))||':  '||'BASE='||strip(base)||'
COMP='||strip(compare);
            output;
    end;
    else do;
            length reason $400;
```

```
          name = ''; attr = ''; base = ''; compare = ''; reason = 'No difference
found.'; output;
    end;
run;

*********************************************;
*** level 2 : compare variable metadata            ***;
*** output : diff_02                                  ***;
*********************************************;
proc sort data = base_02;
    by memname name;
run;

proc transpose data = base_02 out = b(rename = (_NAME_ = attr col1 = base) drop =
_LABEL_ where = (attr not in ('LIBNAME', 'MEMNAME', 'NAME')));
    by memname name;
    var _all_;
run;

proc sort data = compare_02;
    by memname name;
run;

proc transpose data = compare_02 out = c(rename = (_NAME_ = attr col1 = compare)
drop = _LABEL_ where = (attr not in ('LIBNAME', 'MEMNAME', 'NAME')));
    by memname name;
    var _all_;
run;

proc sort data = b;
    by memname name attr;
run;

proc sort data = c;
    by memname name attr;
run;

data process_02;
    length memname name attr $100;
    merge b(in = a) c(in = b);
    by memname name attr;
    base = strip(base);
    compare = strip(compare);
run;

*** To avoid the comparison on non-existing datasets ***;
proc sql noprint;
    create table skip as
    select distinct name as memname from diff_01
    where prim_diff in (1, 2);
quit;
run;

data process_02;
    length memname $100;
    merge process_02(in = a) skip(in = b);
    by memname;
    if a and not b;
run;
*****************************************************;
data process_02;
    set process_02;
    length reason $400;
```

```
    if cmiss(base, compare) = 1 then do;
           if compare = '' then do;
                  prim_diff = 1; ord = 1; reason = 'VARIABLE REMOVED';
           end;
           else if base = '' then do;
                  prim_diff = 2; ord = 1; reason = 'VARIABLE ADDED';
           end;
    end;
    else if cmiss(base, compare) = 0 and base ne compare then do;
           if attr = 'FORMAT'         then do; prim_diff = 6; ord = 1; reason =
'Difference found : Format of Variable'; end;
           if attr = 'FORMATD'        then do; prim_diff = 7; ord = 1; reason =
'Difference found : Number of Format Decimals of Variable'; end;
           if attr = 'FORMATL'        then do; prim_diff = 8; ord = 1; reason =
'Difference found : Format Length of Variable'; end;
           if attr = 'LABEL'          then do; prim_diff = 3; ord = 1; reason =
'Difference found : Variable Label'; end;
           if attr = 'LENGTH'         then do; prim_diff = 4; ord = 1; reason =
'Difference found : Length of Variable'; end;
           if attr = 'TYPE'           then do; prim_diff = 5; ord = 1; reason =
'Difference found : Type of Variable'; end;

           if attr = 'CHARSET'        then do; prim_diff =  9; ord = 1; reason =
'Difference found : Character Set of Variable'; end;
           if attr = 'COMPRESS'       then do; prim_diff = 10; ord = 1; reason =
'Difference found : Compress Routine of Variable'; end;
           if attr = 'ENCRYPT'        then do; prim_diff = 11; ord = 1; reason =
'Difference found : Encryption Routine of Variable'; end;
           if attr = 'INFORMAT'       then do; prim_diff = 12; ord = 1; reason =
'Difference found : Informat of Variable'; end;
           if attr = 'INFORMD'        then do; prim_diff = 13; ord = 1; reason =
'Difference found : Number of Informat Decimals of Variable'; end;
           if attr = 'INFORML'        then do; prim_diff = 14; ord = 1; reason =
'Difference found : Informat Length of Variable'; end;
           if attr = 'JUST'           then do; prim_diff = 15; ord = 1; reason =
'Difference found : Justification of Variable'; end;
           if attr = 'NODUPKEY'       then do; prim_diff = 17; ord = 1; reason =
'Difference found : Sort Option Status - No Duplicate Keys'; end;
           if attr = 'NODUPREC'       then do; prim_diff = 18; ord = 1; reason =
'Difference found : Sort Option Status - No Duplicate Records'; end;
           if attr = 'PROTECT'        then do; prim_diff = 19; ord = 1; reason =
'Difference found : Password Protection Mode'; end;
           if attr = 'SORTED'         then do; prim_diff = 20; ord = 1; reason =
'Difference found : Sorting Status'; end;
           if attr = 'SORTEDBY'       then do; prim_diff = 21; ord = 1; reason =
'Difference found : Position of Variable in the Sorted by Clause'; end;
           if attr = 'TRANSCOD'       then do; prim_diff = 22; ord = 1; reason =
'Difference found : Transcoded Status of Variable'; end;
           if attr = 'VARNUM'         then do; prim_diff = 23; ord = 1; reason =
'Difference found : Sequence Number of Variable in the Dataset'; end;
    end;
run;

proc sort data = process_02 out = diff_02;
    by memname name prim_diff;
    where prim_diff > .;
run;

proc format;
    value $ typ
           '1' = 'Num'
           '2' = 'Char';
run;
```

```
data diff_02;
    set diff_02;
    if prim_diff in (1, 2) and attr ^= 'TYPE' then delete;

    if prim_diff = 9 then delete;
    if prim_diff = 10 then delete;
    if prim_diff = 11 then delete;
    if prim_diff = 15 then delete;
    if prim_diff = 19 then delete;
    if prim_diff = 22 then delete;
    if prim_diff = 23 then delete;

    if attr = 'TYPE' then do;
           base = put(base, typ.);
           compare = put(compare, typ.);
    end;
    length cat $10;
    if prim_diff in (1,2,3,4,5,6,7,8) then cat = 'Major';
    else cat = 'Minor';
run;

%mnodata(inds=diff_02, flag=nodiff_02);

data diff_02;
    if &nodiff_02 eq 'N' then do;
           set diff_02;
           reason = propcase(reason);
           length desc $400;
           desc = strip(attr)||':  '||'BASE='||strip(base)||'
COMP='||strip(compare);
           output;
    end;
    else do;
           length reason $400;
           memname = ''; name = ''; attr = ''; base = ''; compare = ''; reason = 'No
difference found.';output;
    end;
run;

*********************************************;
*** level 3 : compare variable values       ***;
*** output : diff_03    diff_04                        ***;
*********************************************;

*** missing counts check ***;
proc sort data = base_03 out = b(rename = (nmiss = base dsname = memname) drop =
libname);
    by dsname varname;
run;

proc sort data = compare_03 out = c(rename = (nmiss = compare dsname = memname)
drop = libname);
    by dsname varname;
run;

data process_03;
    length memname varname $100;
    merge b(in = a) c(in = b);
    by memname varname;
run;

*** skip ****************************************;
```

```
proc sql noprint;
    create table skip as
    select distinct name as memname from diff_01
    where prim_diff in (1, 2)
    order by name;

    create table skipvar as
    select distinct name as varname, memname from diff_02
    where prim_diff in (1, 2)
    order by memname, varname;
quit;
run;

data process_03;
    merge process_03(in = a) skip(in = b);
    by memname;
    if a and not b;
run;

data process_03;
    merge process_03(in = a) skipvar(in = b);
    by memname varname;
    if a and not b;
run;
**************************************************;
data process_03;
    set process_03;
    length reason $400;
    if nmiss(base, compare) = 1 then do;
           if compare = . then do;
                  prim_diff = 1; ord = 3; reason = 'MISSING FIXED';
           end;
           else if base = . then do;
                  prim_diff = 2; ord = 3; reason = 'MISSING APPEARED';
           end;
    end;
    else if nmiss(base, compare) = 0 then do;
           if input(base, best.) < input(compare, best.) then do; prim_diff = 3; ord
= 3; reason = 'MISSING COUNTS INCREASED'; end;
           if input(base, best.) > input(compare, best.) then do; prim_diff = 4; ord
= 3; reason = 'MISSING COUNTS DECREASED'; end;
    end;
run;

proc sort data = process_03 out = diff_03;
    by memname varname prim_diff;
    where prim_diff > .;
run;

data diff_03;
    set diff_03;
    length cat $10;
    if prim_diff in (1, 2) then cat = 'Major';
    else cat = 'Minor';
run;

%mnodata(inds=diff_03, flag=nodiff_03);

data diff_03;
    length memname varname attr $100 cat $10;
    if &nodiff_03 eq 'N' then do;
           set diff_03;
           attr = 'COUNTS';
```

```
          reason = propcase(reason);
          length desc $400;
          desc = strip(attr)||':  '||'BASE='||strip(put(base, best.))||'
COMP='||strip(put(compare, best.));
          output;
    end;
    else do;
          length reason $400;
          memname = '';
          varname = '';
          base = .;
          compare = .;
          reason = 'No difference found in missing value counts.';
          attr = 'COUNTS';
          output;
    end;
run;

*** check non-missing value ***;

proc sql noprint;
    create table rest as
    select distinct name from diff_01
    where prim_diff not in (1, 2) and name ^= '';
quit;
run;

%mnobs(inds=rest,nobsvar=restds);

data diff_04;
    stop;
run;

%macro comparevalues();

    %do j = 1 %to &restds;
        data _null_;
            set rest;
            if _n_ eq &j. then do;
                call symput('dsname',strip(name));
            end;
        run;

          proc sort data = val_base_&dsname. out = b(keep = dsname varname varval
count rename = (count = base dsname = memname));
                by dsname varname varval;
                where varval not in ('', '.');
          run;

          proc sort data = val_compare_&dsname. out = c(keep = dsname varname
varval count rename = (count = compare dsname = memname));
                by dsname varname varval;
                where varval not in ('', '.');
          run;

          data process_04;
                merge b(in = a) c(in = b);
                by memname varname varval;
                length reason $400;
                if nmiss(base, compare) = 1 then do;
                      if compare = . then do;
                            prim_diff = 1; ord = 2; reason = 'VALUE REMOVED';
                      end;
```

```
                        else if base = . then do;
                               prim_diff = 2; ord = 2; reason = 'VALUE ADDED';
                        end;
                  end;
                  else if nmiss(base, compare) = 0 then do;
                        if input(base, best.) < input(compare, best.) then do;
prim_diff = 3; ord = 2; reason = 'COUNTS INCREASED'; end;
                        if input(base, best.) > input(compare, best.) then do;
prim_diff = 4; ord = 2; reason = 'COUNTS DECREASED'; end;
                  end;
            run;

            data process_04;
                  length memname $100;
                  merge process_04(in = a) skip(in = b);
                  by memname;
                  if a and not b;
            run;

            data process_04;
                  length memname varname $100;
                  merge process_04(in = a) skipvar(in = b);
                  by memname varname;
                  if a and not b;
            run;

            data diff_temp;
                  set process_04;
                  where prim_diff > .;
            run;

            data diff_04;
                  set diff_04 diff_temp;
                  attr = 'COUNTS';
                  length cat $10;
                  if prim_diff in (1, 2) then cat = 'Major';
                  else cat = 'Minor';
            run;
      %end;

      proc sort data = diff_04;
            by prim_diff memname varname;
      run;

%mend comparevalues;

%if &restds >= 1 %then %do;
      %comparevalues();
%end;


%mnodata(inds=diff_04, flag=nodiff_04);

data diff_04;
      length memname varname attr $100 reason $400;
      if &nodiff_04 eq 'N' then do;
            set diff_04;
            reason = propcase(reason);
            length desc $400;
            desc = strip(attr)||':  BASE='||strip(put(base, best.))||'
COMP='||strip(put(compare, best.));
            output;
      end;
```

```
    else do;
           memname = '';
           varname = '';
           varval = '';
           base = .;
           compare = .;
           reason = 'No difference found.';
           attr = 'COUNTS';
           prim_diff = . ;
           ord = .;
           output;
    end;
run;

*** create summary datasets ***;

**** dataset meta level ***;
data list_01;
    length reason $400 cat $10;
    reason = 'DATASET REMOVED';             prim_diff = 1;             cat =
'Major'; printord = 2; output;
    reason = 'DATASET ADDED';               prim_diff = 2;      cat = 'Major';
printord = 1; output;
    reason = 'OBSERVATIONS REMOVED'; prim_diff = 4.1;    cat = 'Major'; printord =
7; output;
    reason = 'OBSERVATIONS ADDED';          prim_diff = 4.2;    cat = 'Major';
printord = 6; output;
    reason = 'NUMBERS OF VARIABLES REDUCED';             prim_diff = 5.1;    cat =
'Major'; printord = 4; output;
    reason = 'NUMBERS OF VARIABLES INCREASED';           prim_diff = 5.2;    cat =
'Major'; printord = 3; output;
    reason = 'LABEL CHANGED';               prim_diff = 6;             cat =
'Major'; printord = 5; output;
run;

proc sort data = diff_01 out = peak_01(drop = attr base compare);
    by prim_diff descending name;
    where ord = 0;
run;

data peak_01(keep = reason dscnt dslist prim_diff);
    if &nodiff_01 eq 'Y' then do;
           set list_01;
           length dslist $400;
           dscnt = 0; dslist = ''; ord = 0;
    end;
    else do;
           set peak_01;
           by prim_diff;
           length dslist $400 prev $100;
           retain dslist prev dscnt;
           if first.prim_diff then do;
                  dscnt = 1;
                  dslist = strip(name);
           end;
           else do;
                  if strip(name) ^= prev then do;
                  dslist = strip(name)||', '||dslist;
                  dscnt = dscnt + 1;
                  end;
           end;
           prev = strip(name);
           if last.prim_diff then output;
```

```
    end;
run;

data peak_01;
    merge peak_01(in = a drop = reason) list_01(in = b);
    by prim_diff;
    if b and not a then dscnt = 0;
run;

proc sort data = peak_01; by printord; run;

*** variable meta level ***;
data list_02;
    length reason $400 cat $10;
    reason = 'VARIABLE REMOVED';                                        cat =
'Major'; prim_diff = 1; printord = 2; output;
    reason = 'VARIABLE ADDED';                                                cat =
'Major'; prim_diff = 2; printord = 1; output;
    reason = 'Label';                                                         cat =
'Major'; prim_diff = 3; printord = 5; output;
    reason = 'Length';                                                        cat =
'Major'; prim_diff = 4; printord = 4; output;
    reason = 'Type';                                                          cat =
'Major'; prim_diff = 5; printord = 3; output;
    reason = 'Format';                                                        cat =
'Major'; prim_diff = 6; printord = 6; output;
    reason = 'Number of Decimals in Format';                    cat = 'Major';
prim_diff = 7; printord = 8; output;
    reason = 'Format Length';                                             cat =
'Major'; prim_diff = 8; printord = 7; output;
/*  reason = 'Character Set';                                             cat =
'Minor'; prim_diff = 9; printord = 12; output;*/
/*  reason = 'Compress Routine';                                          cat =
'Minor'; prim_diff = 10; printord = 13; output;*/
/*  reason = 'Encryption Routine';                                        cat =
'Minor'; prim_diff = 11; printord = 14; output;*/
    reason = 'Informat';                                                      cat =
'Minor'; prim_diff = 12; printord = 9; output;
    reason = 'Number of Decimals in Informat';                  cat = 'Minor';
prim_diff = 13; printord = 11; output;
    reason = 'Informat Length';                                           cat =
'Minor'; prim_diff = 14; printord = 10; output;
/*  reason = 'Justification';                                             cat =
'Minor'; prim_diff = 15; printord = 15; output;*/
    reason = 'Status - No Duplicate Keys';                      cat = 'Minor';
prim_diff = 17; printord = 16; output;
    reason = 'Status - No Duplicate Records';                   cat = 'Minor';
prim_diff = 18; printord = 17; output;
/*  reason = 'Password Protection Mode';                        cat = 'Minor';
prim_diff = 19; printord = 18; output;*/
    reason = 'Sorting Status';                                                cat =
'Minor'; prim_diff = 20; printord = 19; output;
    reason = 'Position in the Sort by Clause';                  cat = 'Minor';
prim_diff = 21; printord = 20; output;
/*  reason = 'Transcoded Status';                                         cat =
'Minor'; prim_diff = 22; printord = 21; output;*/
run;

proc sort data = diff_02 out = peak_02(drop = attr base compare);
    by prim_diff descending memname;
    where ord = 1;
run;
```

```
data peak_02(keep = reason dslist dscnt prim_diff);
    if &nodiff_02 eq 'Y' then do;
           set list_02;
           length dslist $400;
           dscnt = 0; dslist = ''; ord = 1;
    end;
    else do;
           set peak_02;
           by prim_diff;
           length dslist $400 prev $100;
           retain dslist prev dscnt;
           if first.prim_diff then do;
                  dslist = strip(memname);
                  dscnt = 1;
           end;
           else do;
                  if strip(memname) ^= prev then do;
                  dslist = strip(memname)||', '||dslist;
                  dscnt = dscnt + 1;
                  end;
           end;
           prev = strip(memname);
           if last.prim_diff then output;
    end;
run;

data peak_02;
    merge peak_02(in = a drop = reason) list_02(in = b);
    by prim_diff;
    if b and not a then dscnt = 0;
run;

proc sort data = peak_02; by printord; run;

*** Value meta level: non-missing ***;
data list_03;
    length reason $400 cat $10;
    prim_diff = 1; printord = 2; reason = 'VALUE REMOVED';      cat = 'Major';
output;
    prim_diff = 2; printord = 1; reason = 'VALUE ADDED'; cat = 'Major'; output;
    prim_diff = 3; printord = 4; reason = 'COUNTS INCREASED';   cat = 'Minor';
output;
    prim_diff = 4; printord = 3; reason = 'COUNTS DECREASED';   cat = 'Minor';
output;
run;


proc sort data = diff_04 out = peak_03(drop = attr base compare);
    by prim_diff descending memname;
    where ord = 2;
run;

data peak_03(keep = reason dslist dscnt prim_diff cat);
    if &nodiff_04 eq 'Y' then do;
           set list_03;
           length dslist $400;
           dscnt = 0; dslist = ''; ord = 1;
    end;
    else do;
           set peak_03;
           by prim_diff;
           length dslist $400 prev $100;
           retain dslist prev dscnt;
```

```
            if first.prim_diff then do;
                  dslist = strip(memname);
                  dscnt = 1;
            end;
            else do;
                  if strip(memname) ^= prev then do;
                  dslist = strip(memname)||', '||dslist;
                  dscnt = dscnt + 1;
                  end;
            end;
            prev = strip(memname);
            if last.prim_diff then output;
    end;
run;

data peak_03;
    merge peak_03(in = a drop = reason) list_03(in = b);
    by prim_diff;
    if b and not a then dscnt = 0;
run;

proc sort data = peak_03; by printord; run;

*** value meta level : missing ***;
data list_04;
    length reason $400 cat $10;
    prim_diff = 1; printord = 2; reason = 'MISSING FIXED';
    cat = 'Major'; output;
    prim_diff = 2; printord = 1; reason = 'MISSING APPEARED';                cat =
'Major'; output;
    prim_diff = 3; printord = 3; reason = 'MISSING COUNTS INCREASED';        cat =
'Minor'; output;
    prim_diff = 4; printord = 4; reason = 'MISSING COUNTS DECREASED';        cat =
'Minor'; output;
run;

proc sort data = diff_03 out = peak_04(drop = attr base compare);
    by prim_diff descending memname;
    where ord = 3;
run;

data peak_04(keep = reason dslist dscnt prim_diff);
    if &nodiff_03 eq 'Y' then do;
            set list_04;
            length dslist $400;
            dscnt = 0; dslist = ''; ord = 2;
    end;
    else do;
            set peak_04;
            by prim_diff;
            length dslist $400 prev $100;
            retain dslist prev dscnt;
            if first.prim_diff then do;
                  dslist = strip(memname);
                  dscnt = 1;
            end;
            else do;
                  if strip(memname) ^= prev then do;
                  dslist = strip(memname)||', '||dslist;
                  dscnt = dscnt + 1;
                  end;
            end;
            prev = strip(memname);
```

```
           if last.prim_diff then output;
    end;
run;

data peak_04;
    merge peak_04(in = a drop = reason) list_04(in = b);
    by prim_diff;
    if b and not a then dscnt = 0;
run;

proc sort data = peak_04; by printord; run;

*** general info ***;

data diff_all(drop = attr basen comparen);
    length memname varname  $100 desc $400;
    set diff_01(rename = (name = memname) drop = attr base compare)
           diff_02(rename = (name = varname) drop = attr base compare)
           diff_03(rename = (base = basen compare = comparen) )
           diff_04(rename = (base = basen compare = comparen));
run;

data diff_all;
    set diff_all;
    length varclass $20;
    if ord in (2, 3) then do;
                  if index(varname, 'STUDYID') > 0 then varclass = 'STUDYID';
           else if index(varname, 'USUBJID') > 0 then varclass = 'USUBJID';

           else if index(varname, 'DOMAIN') > 0 then varclass = 'DOMAIN';

           else if index(varname, 'PARAMCD') > 0 then varclass = 'PARAMCD';
           else if index(varname, 'SUBJID') > 0 then varclass = 'SUBJID';
           else if index(varname, 'METHOD') > 0 then varclass = '--METHOD--';
           else if index(varname, 'SITE') > 0 then varclass = '--SITE--';
           else if index(varname, 'RIND') > 0 then varclass = '--RIND--';
           else if index(varname, 'VISIT') > 0 then varclass = '--VISIT--';
           else if index(varname, 'PARAM') > 0 then varclass = '--PARAM--';
           else if index(varname, 'EPOCH') > 0 then varclass = '--EPOCH--';
           else if index(varname, 'STRES') > 0 then varclass = '--STRES--';
           else if index(varname, 'ORRES') > 0 then varclass = '--ORRES--';

           else if varname = 'IDVARVAL' then varclass = 'IDVARVAL';
           else if varname = 'IDVAR' then varclass = 'IDVAR';
           else if varname = 'QLABEL' then varclass = 'QLABEL';
           else if varname = 'QVAL' then varclass = 'QVAL';
           else if varname = 'QNAM' then varclass = 'QNAM';

           else if index(varname, 'TOX') > 0 then varclass = '--TOX--';
           else if substr(varname, 1, 3) = 'AGE' then varclass = 'AGE--';
           else if substr(varname, 1, 3) = 'SEX' then varclass = 'SEX--';
           else if substr(varname, 1, 3) = 'ARM' then varclass = 'ARM--';
           else if substr(varname, 1, 4) = 'RACE' then varclass = 'RACE--';
           else if substr(varname, 1, 6) = 'ETHNIC' then varclass = 'ETHNIC--';
           else if substr(varname, 1, 7) = 'COUNTRY' then varclass = 'COUNTRY--';

           else if length(varname) >= 6 and substr(varname, length(varname)-5) =
'MODIFY' then varclass = '--MODIFY';
           else if length(varname) >= 6 and substr(varname, length(varname)-5) =
'TESTCD' then varclass = '--TESTCD';
    /*     else if length(varname) >= 6 and substr(varname, length(varname)-5) =
'HLGTCD' then varclass = '--HLGTCD';*/
```

```
          else if length(varname) >= 6 and substr(varname, length(varname)-5) =
'BODSYS' then varclass = '--BODSYS';

    /*     else if length(varname) >= 5 and substr(varname, length(varname)-4) =
'SOCCD' then varclass = '--SOCCD';*/
    /*     else if length(varname) >= 5 and substr(varname, length(varname)-4) =
'LLTCD' then varclass = '--LLTCD';*/
    /*     else if length(varname) >= 5 and substr(varname, length(varname)-4) =
'HLTCD' then varclass = '--HLTCD';*/
          else if length(varname) >= 5 and substr(varname, length(varname)-4) =
'DECOD' then varclass = '--DECOD';
          else if length(varname) >= 5 and substr(varname, length(varname)-4) =
'OCCUR' then varclass = '--OCCUR';
          else if length(varname) >= 5 and substr(varname, length(varname)-4) =
'PRESP' then varclass = '--PRESP';

          else if length(varname) >= 4 and substr(varname, length(varname)-3) =
'TEST' then varclass = '--TEST';
          else if length(varname) >= 4 and substr(varname, length(varname)-3) =
'TERM' then varclass = '--TERM';
          else if length(varname) >= 4 and substr(varname, length(varname)-3) =
'STAT' then varclass = '--STAT';
          else if length(varname) >= 4 and substr(varname, length(varname)-3) =
'HLGT' then varclass = '--HLGT';

          else if length(varname) >= 3 and substr(varname, length(varname)-2) =
'SEQ' then varclass = '--SEQ';
          else if length(varname) >= 3 and substr(varname, length(varname)-2) =
'NAM' then varclass = '--NAM';
          else if length(varname) >= 3 and substr(varname, length(varname)-2) =
'LOC' then varclass = '--LOC';
          else if length(varname) >= 3 and substr(varname, length(varname)-2) =
'SOC' then varclass = '--SOC';
          else if length(varname) >= 3 and substr(varname, length(varname)-2) =
'LLT' then varclass = '--LLT';
          else if length(varname) >= 3 and substr(varname, length(varname)-2) =
'HLT' then varclass = '--HLT';
          else if length(varname) >= 3 and substr(varname, length(varname)-2) =
'DTC' then varclass = '--DTC';
          else if length(varname) >= 3 and substr(varname, length(varname)-2) =
'DTF' then varclass = '--DTF';
          else if length(varname) >= 3 and substr(varname, length(varname)-2) =
'TRT' then varclass = '--TRT';
          else if length(varname) >= 3 and substr(varname, length(varname)-2) =
'CAT' then varclass = '--CAT';
          else if length(varname) >= 3 and substr(varname, length(varname)-2) =
'DTH' then varclass = '--DTH';

          else if length(varname) >= 2 and substr(varname, length(varname)-1) =
'DT' then varclass = '--DT';
          else if length(varname) >= 2 and substr(varname, length(varname)-1) =
'DY' then varclass = '--DY';
          else if length(varname) >= 2 and substr(varname, length(varname)-1) =
'RF' then varclass = '--RF';
          else if length(varname) >= 2 and substr(varname, length(varname)-1) =
'FL' then varclass = '--FL';
          else if length(varname) >= 2 and substr(varname, length(varname)-1) =
'PT' then varclass = '--PT';
          else if length(varname) >= 2 and substr(varname, length(varname)-1) =
'CD' then varclass = '--CD';
          else if length(varname) >= 2 and substr(varname, length(varname)-1) =
'ID' then varclass = '--ID';
```

```
          else if length(varname) >= 2 and substr(varname, length(varname)-1) =
'HI' then varclass = '--HI';
          else if length(varname) >= 2 and substr(varname, length(varname)-1) =
'LO' then varclass = '--LO';

          else if substr(varname, 1, 3) = 'TRT' then varclass = 'TRT--';
          else if index(varname, 'SER') > 0 then varclass = '--SER--';
          else if index(varname, 'REL') > 0 then varclass = '--REL--';
          else if index(varname, 'OUT') > 0 then varclass = '--OUT--';
          else if index(varname, 'ACN') > 0 then varclass = '--ACN--';
          else if index(varname, 'CAT') > 0 then varclass = '--CAT--';
          else if index(varname, 'DTH') > 0 then varclass = '--DTH--';
          else if index(varname, 'DOS') > 0 then varclass = '--DOS--';
          else varclass = 'Other';
    end;
run;

proc sort data = diff_all; by memname ord cat; run;

proc sql noprint;
    select count(distinct name) into:cds from process_01;
    select count(distinct memname) into:dds11 from diff_all where cat = 'Major';
    select count(distinct memname) into:dds12 from diff_all where cat = 'Minor';

    create table dslist as select distinct name as memname from process_01;

    create table diffdslist as
    select distinct memname
    from diff_all
    where reason not in ('Difference Found : File Size Of Dataset', 'Difference
Found : Last Modified Date/Time Of Dataset');
quit;run;

%put &cds &dds11 &dds12;

data nodiff;
    merge dslist(in = a) diffdslist(in = b);
    by memname;
    if a and not b;
run;

data diffdslist;
    set diffdslist;
    cnt = _n_;
run;

proc sql noprint;
    select count(distinct memname) into:all_diff from diffdslist;
    select count(distinct memname) into:all_nodiff from nodiff;
quit;run;

data peak_0;
    length reason dscnt $400;
    reason = '# of Datasets Compared :'; dscnt = strip(put(&cds, 8.0)); output;
    reason = '# of Datasets with Difference(s) :'; dscnt = strip(put(&all_diff,
8.0))||' ('||strip(put(&all_diff/&cds * 100, 8.0))||'%)'; output;
    reason = '# of Datasets without Difference(s) :'; dscnt =
strip(put(&all_nodiff, 8.0))||' ('||strip(put(&all_nodiff/&cds * 100, 8.0))||'%)';
output;
    reason = '# of Datasets with Major Difference(s) :'; dscnt = strip(put(&dds11,
8.0))||' ('||strip(put(&dds11/&cds * 100, 8.0))||'%)'; output;
    reason = '# of Datasets with Minor Difference(s) :'; dscnt = strip(put(&dds12,
8.0))||' ('||strip(put(&dds12/&cds * 100, 8.0))||'%)'; output;
```

```
run;

**************************;
data timer;
    set timer; output;
    dt = datetime();output;
run;

data header(keep = col1 col2);
    length col1 $40 col2 $200;
    set timer end = last;
    retain start end;
    if _n_ = 1 then start = dt;
    if _n_ = 2 then end = dt;
    if last then do;
           col1 = 'Base lib:'; col2 = strip("&basepath"); ord = 1; output;
           col1 = 'Compare lib:'; col2 = strip("&comparepath"); ord = 2; output;
           col1 = 'Avoid datasets:'; col2 = strip("&avoiddslist"); ord = 2.1;
output;
           col1 = 'Avoid variables:'; col2 = strip("&avoidvarlist"); ord = 2.2;
output;
           col1 = 'Start date:'; col2 = strip(put(start, DTWKDATX.)); ord = 3;
output;
           col1 = 'Start time:'; col2 = strip(put(timepart(start), TIME.)); ord = 4;
output;
           col1 = 'End date:'; col2 = strip(put(end, DTWKDATX.)); ord = 5; output;
           col1 = 'End time:'; col2 = strip(put(timepart(end), TIME.)); ord = 6;
output;
           col1 = 'Time elapsed:'; col2 = put((end - start),5.0)||' s'; ord = 7;
output;
    end;
run;

proc printto;
run;

data _null_;
    call execute(cat("systask command 'copy
",'"',"&SAS_EXECFILEPATH.\&logn..log",'"
"',"&SAS_EXECFILEPATH.\&logn._tmp.log",'"',
                     "' wait status=copyfl;"));
    stop;
run;

filename inlog "&logn._tmp.log";

** read in log and keep system message that begin with note/info/warning/error **;
data msg(keep=line);
    length line $1000;
    infile inlog length=reclen;
    input @1 line $varying200. reclen ;

    if line ne '' ;
    if lowcase(scan(line,1,' ,:()')) in ('error', 'warning') ;
    line = 'Error(s)/Warning(s) appeared in the log';
run;

data _null_;
    call execute(cat("systask command 'del
",'"',"&SAS_EXECFILEPATH.\&logn._tmp.log",'"',"';"));
    stop;
run;
```

```
**********Excel  DDE ************************;
*** open excel template ***;
%put &SAS_EXECFILEPATH;

options noxwait noxsync;
x %str(%')%unquote("&templatepath")%str(%');
filename commands dde "Excel|system";

*** header of summary sheet***;
filename tab dde "Excel|summary!r2c1:r10c2";

data _null_;
    set header;
    file tab notab lrecl = 7000;
    put col1 '09'x col2 '09'x;
run;

%mnodata(inds=msg, flag=noerror);

%if &noerror eq 'N' %then %do;
    filename tab dde "Excel|summary!r6c3:r6c3";
    data _null_;
           set msg;
           file tab notab lrecl = 7000;
           put line '09'x ;
    run;
%end;

*** Summary page***;
filename tab dde "Excel|summary!r14c1:r18c2";

data _null_;
    set peak_0;
    file tab notab lrecl = 7000;
    put reason '09'x dscnt '09'x;
run;

filename tab dde "Excel|summary!r22c1:r30c4";
data _null_;
    set peak_01;
    file tab notab lrecl = 7000;
    put cat '09'x reason '09'x dslist '09'x dscnt '09'x;
run;

filename tab dde "Excel|summary!r34c1:r54c4";
data _null_;
    set peak_02;
    file tab notab lrecl = 7000;
    put cat '09'x reason '09'x dslist '09'x dscnt '09'x;
run;

filename tab dde "Excel|summary!r58c1:r61c4";
data _null_;
    set peak_03;
    file tab notab lrecl = 7000;
    put cat '09'x reason '09'x dslist '09'x dscnt '09'x;
run;

filename tab dde "Excel|summary!r62c1:r66c4";
data _null_;
    set peak_04;
    file tab notab lrecl = 7000;
```

```
    put cat '09'x reason '09'x dslist '09'x dscnt '09'x;
run;

filename commands dde "Excel|system" ;
data _null_;
    file commands;
    put '[workbook.next()]';
    put '[workbook.insert(3)]';
    put '[workbook.move("macro1", "result_template.xls", 4)]';
run;

*** worksheets***;
%mnobs(inds=diffdslist,nobsvar=difflistobs);
%do i = 1 %to &difflistobs;
    proc sql noprint;
           select memname into:pick from diffdslist where cnt = &i;
           create table picker as
           select * from diff_all where memname = "&pick" order by cat, ord,
varname, prim_diff;
    quit;run;

    %put &pick;

    %let tabno=%eval(&i + 1);
    data _null_;
           file commands;
           put '[workbook.activate("template")]';
           put '[workbook.copy("template", "result_template.xls", '"&tabno"')]';
    run;

    %mnobs(inds=picker,nobsvar=pickerobs);
    %let rend=%eval(&pickerobs + 3);
    filename tab dde "Excel|template (2)!r2c1:r&rend.c11";
    data _null_;
           set picker;
           by cat ord;
           if ord = 3 then ord = 2;
           file tab notab lrecl = 7000;
           put ord '09'x cat '09'x varname '09'x reason '09'x desc '09'x varval
'09'x varclass '09'x;
    run;

    filename xlmacro dde 'Excel|macro1!r1c1:r20c1' notab lrecl=200;
    data _null_;
           file xlmacro;
           put '=workbook.name("template (2)", "'"%trim(&pick)"'")';
           put '=halt(true)';
           put '!dde_flush';
           file commands;
           put '[run("macro1!r1c1")]';
    run;
%end;

*** save excel file ***;
filename commands dde "Excel|system" ;
data _null_;
    file commands;
    put '[workbook.activate("summary")]';
    put '[save.as("'"&outfile"'")]';
run;

filename commands clear;
```

```
%mend mlibcompare;
```

APPENDIX B

Excel Template - Sheet 1 : Summary Sheet

Summary:			
LEVEL - 0	**Differences Found in DATASET Level Metadata**		
Significance	**Details**	**List of Datasets with difference(s)**	**Total (datasets)**
LEVEL - 1	**Differences Found in VARIABLE Attributes**		
Significance	**Details**	**List of Datasets with difference(s)**	**Total (datasets)**

LEVEL - 2	**Differences Found in VALUE Level Metadata**		
Significance	**Details**	**List of Datasets with difference(s)**	**Total (datasets)**
Notes:			
(1)	This report only summarizes the information of metadata differences.		
(2)	For detail information of different level compare result, please refer to **different tabs**.		
(3)	A clean compare result doesn't necessarily mean a completely matching. Certain differences will not be detected such as Observations are switched/shifted or Values in certain variable(s) are switched/shifted.		

Excel Template - Sheet 2 : Individual Data Set Sheet

Level	Significance	Variable Name	Description	Details	Value	Filter