SESUG Paper 177-2018

A User Defined SDTM Data Quality Checking and Tracking System

Zemin Zeng and Yunzhi Ling, Sanofi, Bridgewater, NJ

ABSTRACT

The quality of Study Data Tabulation Model (SDTM) datasets plays a crucial role in clinical studies and regulatory submissions. Inspired by Pinnacle 21 report, a similar SDTM quality checking system with additional built-in features to effectively monitor database cleaning and SDTM mapping quality progress has been developed. The checking system consists of collecting checking rules in Excel, writing small SAS® macros to check SDTM datasets, and using SAS data steps and ExcelXP tagset to generate issue summary report in Excel file. This user defined checking system can be easily maintained and expanded through SAS programs creation and modification, and can be shared from one study to another. The paper describes the development steps of the SDTM quality checking and tracking system in details.

INTRODUCTION

In Pharmaceutical industry, the data collected in CRFs and by external vendors is first mapped into Study Data Tabulation Model (SDTM) domains following CDISC guidelines. SDTM datasets are then served as the source datasets for downstream Analysis Data Model (ADaM) datasets and Table/Listing/Graph (TLG) creation, and are included in submissions to health authorities such as FDA and the Pharmaceuticals and Medical Devices Agency (PMDA). As a result, SDTM data quality is critical and requires closely monitor and checks.

Because SDTM datasets are transformed from raw data, during study conduct while data cleaning by clinical data management is still on going, SDTM data issues contain a mixture of database and SDTM mapping issues. Thus, SDTM overall quality is controlled by both clinical data management's proper database quality checking and SDTM creation team's correct mapping algorithms and programming. To reach SDTM quality control goal, it is desired to have an effective SDTM quality checking and tracking system to identify issues, separate them into different categories, and keep tracking the progress during study conduct until formal deliverables.

Pinnacle 21 is a commercial tool to check SDTM data, mainly on structure, consistency among different domains and mostly useful based on cleaned data, when approaching database lock (DBL) for example. During study conduct, Pinnacle 21 report can be very lengthy if SDTM contains lots of database issues, and is time consuming to manually analyze and add explanation to each finding for recording and communication purpose; additionally, it can't track changes from one report to another. On the other hand, SDTM developers and users often have their own SAS programs to check SDTM quality, with checking results usually scattered in separate SAS logs and outputs, but not in a consolidated summary report.

For improvement, we developed a SDTM quality checking system to consolidate potential issues by writing small SAS macros to check and produce issue summary report with status tracking added, to effectively monitor database cleaning and SDTM mapping quality. This SDTM checking system can be run for each data transfer to capture both database and mapping issues timely, produce an issue summary, and track the changes between two data transfers' summary reports.

SDTM DATA QUALITY CHECKING SYSTEM CREATION – A STRUCTURED APPROACH

The SDTM data quality checking system contains 3 sections:

- Consolidate SDTM quality checking rules in an excel file
- Develop SAS macros to tackle each rule
- Run a central SAS program to execute the checking system

Next, details for each section will be described with SAS code provided. The sample checks used in the paper are simple and mainly for illustration purpose.

PRE-SPECIFIED RULES TO CHECK SDTM DATASET QUALITY

First, all the checking rules (including industry/company standards, and study specific) are collected and archived in a central Excel file, as shown in **Table 1** below. The checking rules are organized by domains as shown in column A, with related checking variables listed in column B. Column C describes the checking rules. In column D, the specified issues are categorized into either data issues or mapping issues to facilitate cross-functional communications: data issues need data management to check the database; mapping issues usually require SDTM creation team to check SDTM mapping algorithms and fix SDTM mapping programs accordingly.

In column E, the checking rules are further evaluated and classified as either study specific (Y) or general (N). Non-study specific rules can be developed, accumulated, and shared from one study to another. Study specific checks can be added by individual programmers who identified problems during development and validation of SDTM datasets. This classification makes the migration of the checking system easier among different clinical studies.

Column F specifies the SAS macros developed for the checking rules, with a name convention of %ck_SDTM domain name (e.g., %ck_dm to check potential issues in DM domain). For across multiple domains check, it's named using the dominant domain. For example, row #4 is a cross-checking of first dose date in DM vs. that in EX, and it is included in %ck_dm following the convention. All study specific checks are put into a single macro called %ck_study, e.g., row #22 to check patient re-consented flag vs. new protocol version, and row #26 for comparing treatment completion date in DS against treatment completion flag in DM.

Table 1: Predefined rules for programming checks

	A	B	C	D	E	F
1	Domain	Variable	Rules	Category	Study Specific	Macros
2	DM	USUBJID	No Duplicate USUBJID	data issue	N	%ck_dm
3	DM	RACEOTH	RACEOTH should not be missing if RACE=other	data issue	N	%ck_dm
4	DM/DS	ICDTC	Inform consent date match with information from DS	mapping	N	%ck_dm
21	DM/EX	RFSTDTC	RFSTDTC in DM consistent with the first dose date from EX	mapping	N	%ck_dm
22	DS	DS1RCANY/DS1PV	Patient re-consented flagged as Y, expect Protocol version	data issue	Y	%ck_study
23	DS	DSCAT/DSTERM	Create death record if AEDTHDT not missing	mapping	N	%ck_ds
24	DS	DSCAT/DSTERM	Each patient should have no more than one records per event	mapping	N	%ck_ds
25	DS/DM/AE	DSCAT/DTHDTC	DEATH inform in DS, DM and AE consistent	mapping	N	%ck_ds
26	DS/DM	DSSCAT/COMPLT	If DSSCAT='END OF TREATMENT' and DSDECOD='COMPLETED', then DS4ETDT in DM should not ' ' and DS4WTDT should be ' '; If DSSCAT='END OF TREATMENT' and DSDECOD ne 'COMPLETED', then DS4ETDT should be ' ' and DS4WTDT should not be ' ';	data issue	Y	%ck_study
27	EX	EXOCCUR	EXOCCUR='Y' expect EXSTDTC/EXNBX not missing	data issue	N	%ck_ex

DEVELOP SAS MACROS TO CHECK AGAINST EACH RULE

After the checking rules are collected and consolidated, corresponding SAS macros are developed to check all the rules domain by domain. Within a checking macro, one block of SAS codes is to check one rule and output a temporary dataset to record domain name, key variables used for checking, rules description, report value, category and number of issues. At the end of the macro, all issue occurrence datasets are set together for further analyses.

Example 1 below shows the outline of %ck_dm macro, with detailed rule #1 and rule #20 checking codes included. %ck_dm macro output a result dataset ck_dm which contains all issue records based on all checking rules for DM domain.

Example 1 General checking: SAS Sample Code in %ck_dm macro:

```
%macro ck_dm;
/*Below are sample SAS code to output the checking result dataset ck_dm(say 20 general checking rules in ck_dm macro)*/
/***DM domain Rule 1: No Duplicate USUBJID***/
  data tmp; set sdd.dm;
    by usubjid; if not first.usubjid;
  run;
  proc sort data=tmp nodupkey; by usubjid;run;
  proc sql;   select count(*) into: nobs from tmp; quit;
  data dm_1;
    length DATASET VARIABLE $15 RULES $200 Report_value $200 Category $20 Number_of_Issue $10;
    set tmp ;
    DATASET='DM'; variable="USUBJID"; RULES="No Duplicate USUBJID";
    report_value="USUBJID = "||strip(usubjid); category="Data Issue"; number_of_issue="&nobs";
    keep DATASET variable rules report_value category number_of_issue;
  run;
**the code for other 18 rules- omitted here. Output dataset dm_2, .... dm_19;
*****************************************************************;
/***DM domain Rule 20: RFSTDTC in DM consistent with the first dose date from EX***/
  data ex; set rdbs.ex;
    where excat ="INVESTIGATIONAL PRODUCT ADMINISTRATION" and exoccur^="N" and EXNBX ^="" and exstdtc ^="";
  run;
  data rfxstdt (keep=usubjid exstdtc); set ex; by usubjid; if first.usubjod; run;
  data tmp; merge rfxstdt(in=a) sdd.dm(keep=usubjid rfstdtc in=b);
    by usubjid; if a and b and exstdtc ne rfstdtc; run;
  proc sql;   select count(*) into: nobs from tmp; quit;
  data dm_20;
    length DATASET VARIABLE $15 RULES $200 Report_value $200 Category $20 Number_of_Issue $10;
    set tmp ;
    DATASET='DM'; variable="RFSTDTC"; RULES="RFSTDTC in DM consistent with the first dose date from EX";
    report_value="USUBJID = "||strip(usubjid) || ", RFSTDTC=" || strip(RFSTDTC) || ", EXSTDTC=" || strip(EXSTDTC);
    category="Data Issue"; number_of_issue="&nobs";
    keep DATASET variable rules report_value category number_of_issue;
  run;

*Set each rule outputs dm_1 - dm_20 together to output ck_dm for macro %ck_dm;
  data ck_dm; set dm_: ; run;
%mend ck_dm;
```

Output 1 below shows some sample records of ck_dm dataset. To save space, only the checking results of Rule 1 (No Duplicate USUBJID) are listed. There are five subjects having duplicated USUBJID in DM domain, so five records are output into ck_dm. Each row presents one duplicated USUBJID shown in Report_value column, with "5" stored in Number of Issue column to indicate the magnitude of this issue.

Output 1 Sample records from dataset ck_dm:

	DATASET	VARIABLE	RULES	Report_value	Category	Number_of_Issue
1	DM	USUBJID	No Duplicate USUBJID	USUBJID = 013579-032-001-101	Data Issue	5
2	DM	USUBJID	No Duplicate USUBJID	USUBJID = 013579-032-001-102	Data Issue	5
3	DM	USUBJID	No Duplicate USUBJID	USUBJID = 013579-032-001-103	Data Issue	5
4	DM	USUBJID	No Duplicate USUBJID	USUBJID = 013579-032-001-104	Data Issue	5
5	DM	USUBJID	No Duplicate USUBJID	USUBJID = 013579-032-001-105	Data Issue	5

Following the same logic of %ck_dm built-up, study specific checking rules are created and saved into a single SAS macro called %ck_study, as shown in **Example 2**. Again, one block of SAS codes is for one rule, with all issue occurrences saved into one temporary dataset, study_i. In the end, all study specific checking results datasets are set together into one dataset called ck_study. **Output 2** below lists some sample records of ck_study dataset.

Example 2 Study specific checking: Sample SAS Code in %ck_study macro:

```
%macro ck_study;
*Sample SAS code in %ck_study to check study specific rules;

**Study specific check Rule 1: Patient re-consented flagged as Y, expect Protocol version;
data tmp; set adsd.ds;
  where DS1RCANY ='Y' and DS1PV='';
  keep usubjid DS1RCANY DS1PV;
run;

proc sql noprint;   select count(*) into: nobs from tmp; quit;

data study_1;
  length DATASET VARIABLE $15 RULES $200 Report_value $200 Category $20 Number_of_Issue $10;
  set tmp ;
  DATASET='DS'; variable="DS1RCANY/DS1PV"; RULES='Patient re-consented flagged as Y, expect Protocol version';
  report_value="USUBJID = "||strip(usubjid)||", DS1RCANY = "||strip(DS1RCANY)||
               ", DS1PV = "||strip(DS1PV);
  category="Data Issue"; number_of_issue="&nobs";
  keep DATASET variable rules report_value category number_of_issue;
run;

/*Other Study specified rule checks ********* */

*Set all study_xx together to output ck_study for macro %ck_study;
data ck_study; set study_: ; run;

%mend ck_study;
```

Output 2 Sample records from dataset ck_study:

STUDY_1

Filter and Sort | Query Builder | Data | Describe | Graph | Analyze | Export | Send To

	DATASET	VARIABLE	RULES	Report_value	Category	Number_of_Issue
1	DS	DS1RCANY/DS1PV	Patient re-consented flagged as Y, expect Protocol version	USUBJID = 013579-032-001-102, DS1RCANY = Y, DS1PV =	Data Issue	4
2	DS	DS1RCANY/DS1PV	Patient re-consented flagged as Y, expect Protocol version	USUBJID = 013579-032-001-103, DS1RCANY = Y, DS1PV =	Data Issue	4
3	DS	DS1RCANY/DS1PV	Patient re-consented flagged as Y, expect Protocol version	USUBJID = 013579-032-001-105, DS1RCANY = Y, DS1PV =	Data Issue	4
4	DS	DS1RCANY/DS1PV	Patient re-consented flagged as Y, expect Protocol version	USUBJID = 013579-032-001-106, DS1RCANY = Y, DS1PV =	Data Issue	4

A CENTRAL SAS PROGRAM TO EXCUTE CHECKING SYSTEM

After all SAS checking macros have been developed, in a central SAS program, there are three steps to excute the SDTM data quality checking system:

- Step 1: Call SAS macros and output checking result dataset;
- Step 2: Analyze the issue status by comparing with previous delivery;
- Step 3: Output checking results into Excel file.

Step 1: Call SAS macros and output checking result dataset.
In the central SAS program, all checking macros are called, with all result datasets ck_: appended together and saved into a permanent dataset for one version of SDTM indicated by &sysdate.

Call checking macros and output final data for reporting:

```
*Call general checking macros;
%ck_dm;
%ck_ds;
%ck_ex;
%ck_ae;
%ck_cm;
*...;

*Call study specific checking macro;
%ck_study;

*Append the general checks by domains (like ck_dm, ck_ds...);
*          and study specific (ck_study) to one final dataset;
data data_check_&sysdate.; set ck_:; run;
```

Step 2: Analyze the issue status by comparing with previous delivery
Next, the most recent checking result dataset is compared with a previously saved version, with comparison result saved into STATUS variable: NEW vs. EXISTING to indicate new issues found in this version vs. previously identfied issues. This status indicator helps the responsible teams to further check the problems and follow up issue resolving.

Analyze issue status:

```
 *To analyze the status by comparing with previous delivery;
proc sort data=data_check_&sysdate. out=current;
   by dataset rules report_value;
run;

 *dateprv is the date of the last data transfer;
 %let dateprv=09DEC17;

data qcd.data_check_&sysdate.;
   merge current(in=a) qcd.data_check_&dateprv (in=b drop=status);
   by dataset rules report_value;
   length status $10;
   if a and b then status="EXISTING";
   else if a and not b then status='NEW';
   if a;
run;
```

Step 3: Output checking results in Excel file

Through SAS data steps and ExcelXP tagset, the full issue checking results and a subset of checking list are sent to a single Excel file containing two different worksheets, as illustrated in below program.

A subset of checking result dataset is created by selecting the first three records for each rule by status (NEW or EXISTING): SEQ is first derived to indicate the position of the issues rocords, then use condition SEQ<=3 to subset the first three records. Next, two ods tagsets.excelxp procedures send the whole issue dataset to "Full list" worksheet and subset issue dataset (final) to "3 examples each rule" worksheet, seperately. Interested users may refer to SAS website: https://support.sas.com/rnd/base/ods/odsmarkup/excelxp_demo.html for detailed examples and syntax usage of ExcelXP tagset.

Output full or partial checking results to Excel file:

```
data issue;
  set qcd.data_check_&sysdate.; SEQ+1;
  by dataset variable rules status;
  if first.status then SEQ=1;
run;

data final;
  set issue(where =(seq<=3)); /*Only output 3 examples for each rule by status*/
run;

*Output dataset into EXCEL;
ods tagsets.excelxp style=printer file="&outpath/DATA_CHECK_RESULT_&sysdate..xls" ;
  *A). Output full list of issues;
    ods tagsets.excelxp options(frozen_headers="yes" autofilter="on" sheet_interval='none'
        sheet_name="Full list" absolute_column_width="10,15,45,15,45,13,13");
    proc print data=qcd.data_check_&sysdate. noobs;
         var dataset variable rules category report_value number_of_issue status;
    run;

  *B). Output 3 examples for each rule for clean patients;
    ods tagsets.excelxp options(frozen_headers="yes" autofilter="on" sheet_interval='none'
        sheet_name="3 examples each rule" absolute_column_width="10,15,45,15,45,13,13");
    options linesize=256;
    proc print data=final noobs;
      var dataset variable rules category report_value number_of_issue status;
    run;
ods tagsets.excelxp close;
```

Table 2.1 below shows the full list of sample checking results, while **Table 2.2** shows up to three examples for each rule by status. The difference between the two tables is the deletion of two records (blue colored in the first table) as only three examples are kept within each rule by status. During study conduct while data cleanning and SDTM mapping are under-development, there are usaully many mixed issues captured by the system, so the full checking list can be very lengthy, the shortened list makes the review easier and serves the purpose. When SDTM quality being improved further, the full list becomes shorter and shorter.

Table 2.1: Sample Output of SDTM checking system-Full list

	A	B	C	D	E	F	G
1	DATASET	VARIABLE	RULES	Category	Report_value	Number_of_Issue	Status
2	AE	AESTDTC	Subject AESTDTC should not be after AEENDTC	Data Issue	USUBJID = 013579-840-003-127, AESEQ = 24, AESTDTC = 2017-06-05T12:04, AEENDTC = 2017-06-05T00:00	1	NEW
3	DM	USUBJID	No Duplicate USUBJID	Data Issue	USUBJID = 013579-032-001-101	5	EXISTING
4	DM	USUBJID	No Duplicate USUBJID	Data Issue	USUBJID = 013579-032-001-102	5	NEW
5	DM	USUBJID	No Duplicate USUBJID	Data Issue	USUBJID = 013579-032-001-103	5	NEW
6	DM	USUBJID	No Duplicate USUBJID	Data Issue	USUBJID = 013579-032-001-104	5	NEW
7	DM	USUBJID	No Duplicate USUBJID	Data Issue	USUBJID = 013579-032-001-105	5	NEW
8	DS	DS1RCANY/DS1PV	Patient re-consented flagged as Y, expect Protocol version	Data Issue	USUBJID = 013579-032-001-102, DS1RCANY = Y, DS1PV =	4	EXISTING
9	DS	DS1RCANY/DS1PV	Patient re-consented flagged as Y, expect Protocol version	Data Issue	USUBJID = 013579-032-001-103, DS1RCANY = Y, DS1PV =	4	EXISTING
10	DS	DS1RCANY/DS1PV	Patient re-consented flagged as Y, expect Protocol version	Data Issue	USUBJID = 013579-032-001-105, DS1RCANY = Y, DS1PV =	4	EXISTING
11	DS	DS1RCANY/DS1PV	Patient re-consented flagged as Y, expect Protocol version	Data Issue	USUBJID = 013579-032-001-106, DS1RCANY = Y, DS1PV =	4	EXISTING

Table 2.2: Sample Output of SDTM checking system – up to 3 examples for each rule by status

	A	B	C	D	E	F	G
1	DATASET	VARIABLE	RULES	Category	Report_value	Number_of_Issue	Status
2	AE	AESTDTC	Subject AESTDTC should not be after AEENDTC	Data Issue	USUBJID = 013579-840-003-127, AESEQ = 24, AESTDTC = 2017-06-05T12:04, AEENDTC = 2017-06-05T00:00	1	NEW
3	DM	USUBJID	No Duplicate USUBJID	Data Issue	USUBJID = 013579-032-001-101	5	EXISTING
4	DM	USUBJID	No Duplicate USUBJID	Data Issue	USUBJID = 013579-032-001-102	5	NEW
5	DM	USUBJID	No Duplicate USUBJID	Data Issue	USUBJID = 013579-032-001-103	5	NEW
6	DM	USUBJID	No Duplicate USUBJID	Data Issue	USUBJID = 013579-032-001-104	5	NEW
7	DS	DS1RCANY/DS1PV	Patient re-consented flagged as Y, expect Protocol version	Data Issue	USUBJID = 013579-032-001-102, DS1RCANY = Y, DS1PV =	4	EXISTING
8	DS	DS1RCANY/DS1PV	Patient re-consented flagged as Y, expect Protocol version	Data Issue	USUBJID = 013579-032-001-103, DS1RCANY = Y, DS1PV =	4	EXISTING
9	DS	DS1RCANY/DS1PV	Patient re-consented flagged as Y, expect Protocol version	Data Issue	USUBJID = 013579-032-001-105, DS1RCANY = Y, DS1PV =	4	EXISTING

CONCLUSION

The detailed steps to develop our SDTM data quality checking system have been described in the paper. The consolidated issue summary report with status tracking flag helps team to effectively monitor SDTM quality and communicate the status changes. Because the individual SAS rule checking macro can be easily developed, maintained, and updated to accomodate SDTMIG up-versioning and study specific checking needs, this checking system has been used in several internal studies and proved to be very efficient and user-friendly. Following the same principle and sample SAS codes, interested readers can create their own system to programmatically check and track SDTM quality.

ACKNOWLEDGMENTS

The authors sincerely thank our programming team members for providing checking rules and developing all the checking macros.

REFERENCES

The CDISC SDTM implementation guide (SDTMIG): http://www.cdisc.org/sdtm

Pinnacle21: https://www.pinnacle21.com/

BASE SAS: The ExcelXP Tagset and Microsoft Excel
https://support.sas.com/rnd/base/ods/odsmarkup/excelxp_demo.html

CONTACT INFORMATION

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

Zemin Zeng, Sanofi, 55C-330A, 55 Corporate Drive, Bridgewater, NJ 08807. Phone: 908-981-3628,
Email: zemin.zeng@sanofi.com

Yunzhi Ling, Sanofi, 55C-330A, 55 Corporate Drive, Bridgewater, NJ 08807. Phone: 908-981-6169,
Email: Yunzhi.ling@sanofi.com