PharmaSUG 2023 - Paper SD-084

A Macro Utility for CDISC Datasets Cross Checking

Chao Su, Merck & Co., Inc., Rahway, NJ, USA

Jaime Yan, Merck & Co., Inc., Rahway, NJ, USA

Changhong Shi, Merck & Co., Inc., Rahway, NJ, USA

ABSTRACT

High-quality data in clinical trials is essential for compliance with Good Clinical Practice (GCP) and regulatory requirements. However, data issues exist in ADaM and SDTM datasets within and between them in most practical studies. In order to identify and clean data issues before database lock (DBL) or other main milestones, a macro is developed for discrepancy cross-checking between ADaM and SDTM datasets during analysis and reporting processes. In this paper, some common data checks among ADaM and SDTM datasets are presented and discussed. The findings are reported in an excel spreadsheet with a friendly interface consisting of a neat summary tab and individual formatted tab for each data issue category. Moreover, the modularized structure provides excellent scalability and flexibility for the user to add a user-defined rule with simple and easy steps. This feature allows the macro to be used far beyond CDISC datasets. User-defined rules can be extended to various data structures and types across therapeutical areas and studies. This utility provides a friendly and flexible way to check and track data issues related to the A&R process accurately and efficiently.

INTRODUCTION

ADaM and SDTM datasets are widely used in the pharmaceutical industry to submit clinical trial data to regulatory agencies such as the FDA. Discrepancies between ADaM/SDTM datasets and the corresponding specifications can occur for various reasons, such as errors in data entry, differences in data coding conventions, variations in how data are collected and processed, or improper updates to the ADaM/SDTM specifications. These discrepancies can impact the quality and accuracy of the data and may require additional review and validation to ensure compliance with regulatory requirements.

It is crucial to have robust data quality control processes to identify and address these issues early in the data collection, preparation, and finalization. Identifying and addressing data issues early can minimize the impact and produce submission-compliant deliverables.

Additionally, it is essential to have a comprehensive testing plan and validation strategy to ensure that the data is accurate and compliant in the whole clinical trial data analysis process.

Nowadays, software like Pinnacle 21 provides the solution to prepare clinical trial data for regulatory submission with a user-friendly interface. However, such software is suitable to finalize the data package with expected standards; in other words, the proper time to use them is when most of the standard datasets and the specs/related documents are ready. What is more, Pinnacle 21 needs to check if the datasets are developed or not. It checks if the datasets comply with CDISC standards, but it does not check the data's content. Therefore, it needs to check if the data are appropriate for the research question or if the datasets are complete. The Analysis & Report programmer will usually check by reviewing the data and looking for outliers, patterns, or other issues.

The clinical trial analysis process occurs over a long period of time. It is a good practice for programmers to catch the discrepancies in every trial development stage as early as possible. Also, besides the standard checking rule specific to each therapy area, the demand exists to use specific logic to check the data. Moreover, checking the discrepancy of the dataset that does not follow the CDISC standard (like some external data in excel format, Etc.) at each stage of the clinical trial data analysis process is required. In such cases, a macro with flexible, customized, modular features that could create an easy-to-read report becomes a good solution. This paper will introduce a macro that meets the above requirement.

MACRO DESIGN

The primary purposes of this macro are:

- Catch the targeted discrepancies, including the discrepancies between standards and datasets and the ones between different datasets.

- Output the user-friendly discrepancies report in excel format.

To meet the first purpose, the macro supports standard file/datasets reading and provides the function to catch the targeted discrepancies.

Instead of creating a comprehensive macro that includes all potential checking items, this macro attempts to provide a framework that takes care of dataset/file reading and report s output. At the same time, the macro provides module functions (sub-macro) as a tool to solve various discrepancies, checking demand flexibly. The user would use/combine the provided function to check different input datasets/files.. At the same time, the user could add their tools (sub-macro) within the main macro to create additional item-checking module and then use the report-creating function provided by the macro to get an excellent report.

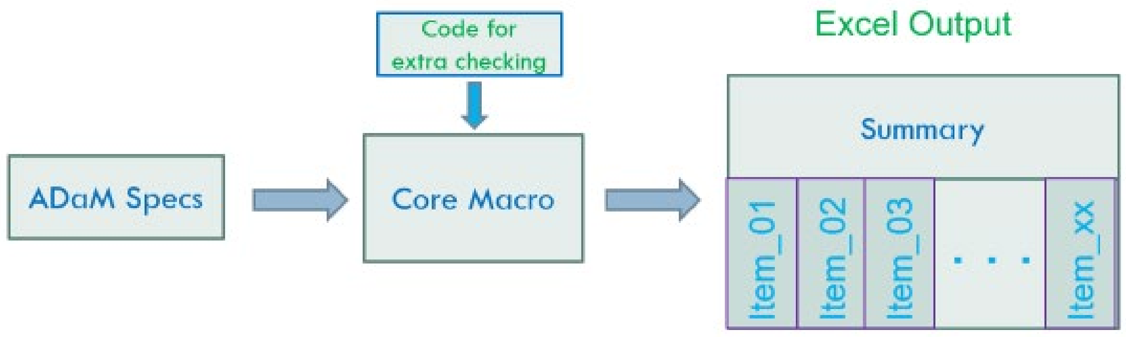


Fig. 1 Macro Diagram

Therefore, the macro/framework includes three main modules as shown at Fig.1 above: Datasets/files input module, default discrepancy checking tools (sub macro), default checking item sample code module, and report creating module.

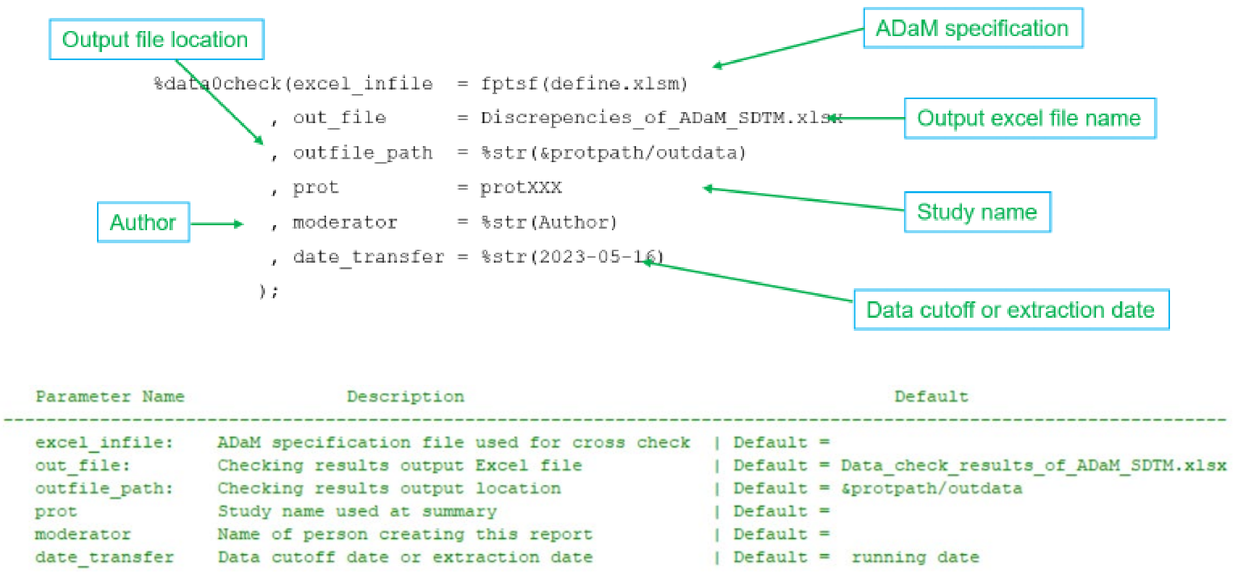
The macro could read data or standard files in both SAS® and excel data formats. Specifically, for the excel file, the macro could automatically recognize the format and support all the general excel file format (like xls, xlsm, Etc.).

In order to check discrepancies and support the future extension of the macro, the macro provides several data-checking functions by encapsulating the default sub-macro as modules within the main macro. For example, the function to check the variable existence, whether specific datasets show correctly, etc. This design pattern lets users easily include their own function macro and achieve their expected checking item.

For the second purpose, the macro provides functions to create an organized excel discrepancy report with contents/background info sheet, user-defined discrepancy category sheet, and items with each discrepancy category sheet.

The user could use the macro directly with default checking items. Also, the user could use the functions provided by the sub macro within the main macro and follow the checking sample to create their checking items, especially when the user has a special discrepancy checking requirement based on their dataset in their therapy area.

Fig. 2 is an example of the macro call. The first parameter of the input section is excel_infile which functions to get the info of the standard file (ADaM specification). The proper input for this parameter includes the file library, file name, and file format/extension. The second and third parameters are output section which include out_file and outfile_path to assign the name and location for the expected output file. The study information is provided by parameter prot. The last two parameters are used to collect the author and date of data like cutoff date or database lock date etc. The info from this section will show in the final excel report.



```
%data0check(excel_infile  = fptsf(define.xlsm)
          , out_file      = Discrepencies_of_ADaM_SDTM.xlsx
          , outfile_path  = %str(&protpath/outdata)
          , prot          = protXXX
          , moderator     = %str(Author)
          , date_transfer = %str(2023-05-16)
         );
```

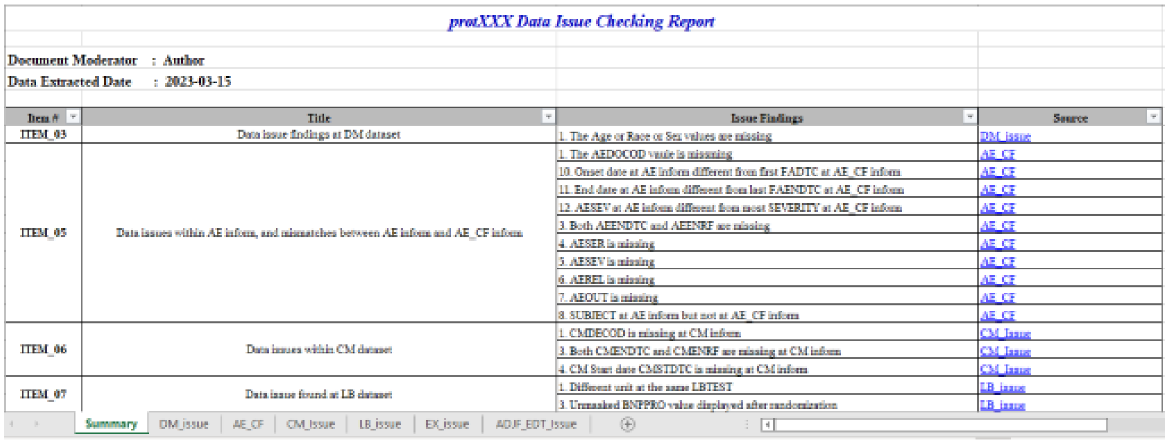
```
 Parameter Name            Description                                           Default
--------------------------------------------------------------------------------------------------
 excel_infile:   ADaM specification file used for cross check  | Default =
 out_file:       Checking results output Excel file            | Default = Data_check_results_of_ADaM_SDTM.xlsx
 outfile_path:   Checking results output location              | Default = &protpath/outdata
 prot            Study name used at summary                    | Default =
 moderator       Name of person creating this report           | Default =
 date_transfer   Data cutoff date or extraction date           | Default =  running date
```

Fig. 2 Sample macro call and description of macro variable

DEFAULT CHECKING ITEMS

The default checking items in the macro focus on using ADaM spec(standard file) and ADaM/SDTM data as input and checking the discrepancies between the spec, SDTM datasets, and ADaM datasets for an ongoing study. However, with the same framework and tools(sub-macro) this macro provides, users could easily create their version for specific checking purposes.

After the specification is ready for this version, the users will follow the specification to identify issues. Different SAS datasets are generated for the various issue categories. These SAS datasets work as the input source for the developed macro to create a single Excel file with one worksheet for each issue category. Another worksheet with a status summary of the data issues is also provided by the macro, as shown in Fig. 3. Within the summary sheet, the hyperlink could direct the reviewer to the specific issue category sheet. As shown in Fig. 3, where the combination of columns "Item" and "Title" indicate the checking categories, the column "Issue Findings" is the description of the data issue; the column "Source" is the hyperlink to the individual worksheet. After clicking the first hyperlink, the sheet transfers to checking item details, as shown in Fig. 4. Some default checking items are listed at Fig. 5.



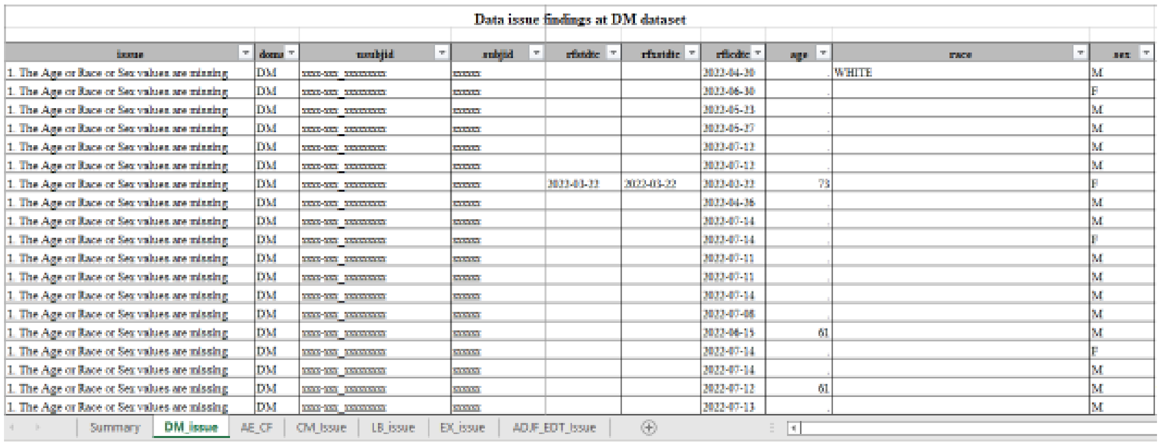
protXXX Data Issue Checking Report

Document Moderator : Author
Data Extracted Date : 2023-03-15

Item #	Title	Issue Findings	Source
ITEM_03	Data issue findings at DM dataset	1. The Age or Race or Sex values are missing	DM_issue
ITEM_05	Data issues within AE inform, and mismatches between AE inform and AE_CF inform	1. The AEDOCOD vaule is missing	AE_CF
		10. Onset date at AE inform different from first FADTC at AE_CF inform	AE_CF
		11. End date at AE inform different from last FAENDTC at AE_CF inform	AE_CF
		12. AESEV at AE inform different from most SEVERITY at AE_CF inform	AE_CF
		3. Both AEENDTC and AEENRF are missing	AE_CF
		4. AESER is missing	AE_CF
		5. AESEV is missing	AE_CF
		6. AEREL is missing	AE_CF
		7. AEOUT is missing	AE_CF
		8. SUBJECT at AE inform but not at AE_CF inform	AE_CF
ITEM_06	Data issues within CM dataset	1. CMDECOD is missing at CM inform	CM_Issue
		3. Both CMENDTC and CMENRF are missing at CM inform	CM_Issue
		4. CM Start date CMSTDTC is missing at CM inform	CM_Issue
ITEM_07	Data issue found at LB dataset	1. Different unit at the same LBTEST	LB_issue
		3. Unmasked BNPPRO value displayed after randomization	LB_issue

Summary | DM_issue | AE_CF | CM_Issue | LB_issue | EX_issue | ADJF_EDT_Issue

Fig. 3 Sample report output for summary sheet

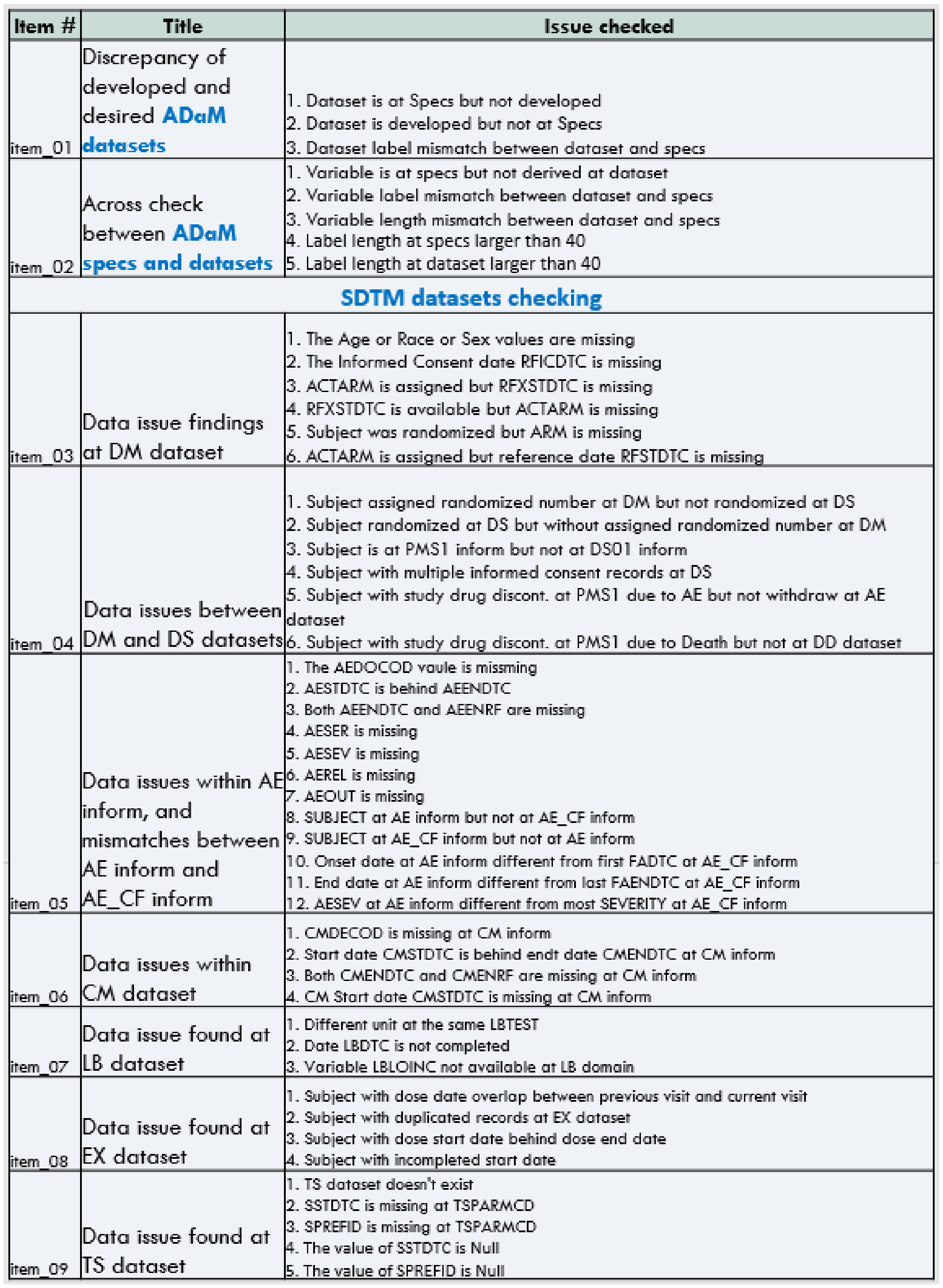


Data issue findings at DM dataset

issue	doma	usubjid	subjid	rfstdtc	rfxstdtc	rficdtc	age	race	sex
1. The Age or Race or Sex values are missing	DM	xxxx-xxx_xxxxxxxx	xxxxxx			2022-04-20	.	WHITE	M
1. The Age or Race or Sex values are missing	DM	xxxx-xxx_xxxxxxxx	xxxxxx			2022-06-30	.		F
1. The Age or Race or Sex values are missing	DM	xxxx-xxx_xxxxxxxx	xxxxxx			2022-05-23	.		M
1. The Age or Race or Sex values are missing	DM	xxxx-xxx_xxxxxxxx	xxxxxx			2022-05-27	.		M
1. The Age or Race or Sex values are missing	DM	xxxx-xxx_xxxxxxxx	xxxxxx			2022-07-12	.		M
1. The Age or Race or Sex values are missing	DM	xxxx-xxx_xxxxxxxx	xxxxxx			2022-07-12	.		M
1. The Age or Race or Sex values are missing	DM	xxxx-xxx_xxxxxxxx	xxxxxx	2022-03-22	2022-03-22	2022-03-22	73		F
1. The Age or Race or Sex values are missing	DM	xxxx-xxx_xxxxxxxx	xxxxxx			2022-04-26	.		M
1. The Age or Race or Sex values are missing	DM	xxxx-xxx_xxxxxxxx	xxxxxx			2022-07-14	.		M
1. The Age or Race or Sex values are missing	DM	xxxx-xxx_xxxxxxxx	xxxxxx			2022-07-14	.		F
1. The Age or Race or Sex values are missing	DM	xxxx-xxx_xxxxxxxx	xxxxxx			2022-07-11	.		M
1. The Age or Race or Sex values are missing	DM	xxxx-xxx_xxxxxxxx	xxxxxx			2022-07-11	.		M
1. The Age or Race or Sex values are missing	DM	xxxx-xxx_xxxxxxxx	xxxxxx			2022-07-14	.		M
1. The Age or Race or Sex values are missing	DM	xxxx-xxx_xxxxxxxx	xxxxxx			2022-07-08	.		M
1. The Age or Race or Sex values are missing	DM	xxxx-xxx_xxxxxxxx	xxxxxx			2022-08-15	61		M
1. The Age or Race or Sex values are missing	DM	xxxx-xxx_xxxxxxxx	xxxxxx			2022-07-14	.		F
1. The Age or Race or Sex values are missing	DM	xxxx-xxx_xxxxxxxx	xxxxxx			2022-07-14	.		M
1. The Age or Race or Sex values are missing	DM	xxxx-xxx_xxxxxxxx	xxxxxx			2022-07-12	61		M
1. The Age or Race or Sex values are missing	DM	xxxx-xxx_xxxxxxxx	xxxxxx			2022-07-13	.		M

Summary | DM_issue | AE_CF | CM_Issue | LB_issue | EX_issue | ADJF_EDT_Issue

Fig. 4 Sample report output for specific category sheet



Item #	Title	Issue checked
item_01	Discrepancy of developed and desired **ADaM datasets**	1. Dataset is at Specs but not developed 2. Dataset is developed but not at Specs 3. Dataset label mismatch between dataset and specs
item_02	Across check between **ADaM specs and datasets**	1. Variable is at specs but not derived at dataset 2. Variable label mismatch between dataset and specs 3. Variable length mismatch between dataset and specs 4. Label length at specs larger than 40 5. Label length at dataset larger than 40
		SDTM datasets checking
item_03	Data issue findings at DM dataset	1. The Age or Race or Sex values are missing 2. The Informed Consent date RFICDTC is missing 3. ACTARM is assigned but RFXSTDTC is missing 4. RFXSTDTC is available but ACTARM is missing 5. Subject was randomized but ARM is missing 6. ACTARM is assigned but reference date RFSTDTC is missing
item_04	Data issues between DM and DS datasets	1. Subject assigned randomized number at DM but not randomized at DS 2. Subject randomized at DS but without assigned randomized number at DM 3. Subject is at PMS1 inform but not at DS01 inform 4. Subject with multiple informed consent records at DS 5. Subject with study drug discont. at PMS1 due to AE but not withdraw at AE dataset 6. Subject with study drug discont. at PMS1 due to Death but not at DD dataset
item_05	Data issues within AE inform, and mismatches between AE inform and AE_CF inform	1. The AEDOCOD vaule is missming 2. AESTDTC is behind AEENDTC 3. Both AEENDTC and AEENRF are missing 4. AESER is missing 5. AESEV is missing 6. AEREL is missing 7. AEOUT is missing 8. SUBJECT at AE inform but not at AE_CF inform 9. SUBJECT at AE_CF inform but not at AE inform 10. Onset date at AE inform different from first FADTC at AE_CF inform 11. End date at AE inform different from last FAENDTC at AE_CF inform 12. AESEV at AE inform different from most SEVERITY at AE_CF inform
item_06	Data issues within CM dataset	1. CMDECOD is missing at CM inform 2. Start date CMSTDTC is behind endt date CMENDTC at CM inform 3. Both CMENDTC and CMENRF are missing at CM inform 4. CM Start date CMSTDTC is missing at CM inform
item_07	Data issue found at LB dataset	1. Different unit at the same LBTEST 2. Date LBDTC is not completed 3. Variable LBLOINC not available at LB domain
item_08	Data issue found at EX dataset	1. Subject with dose date overlap between previous visit and current visit 2. Subject with duplicated records at EX dataset 3. Subject with dose start date behind dose end date 4. Subject with incompleted start date
item_09	Data issue found at TS dataset	1. TS dataset doesn't exist 2. SSTDTC is missing at TSPARMCD 3. SPREFID is missing at TSPARMCD 4. The value of SSTDTC is Null 5. The value of SPREFID is Null

Fig. 5 Default checking items

FEATURE OF THE MACRO

In short, the main features of the macro are:

1. Customization/flexibility: this macro can be customized and tailored to specific data validation checks and business rules, allowing for a more tailored approach to data validation. Also, the format and contents of the report could also be customized.

2. Automation: this macro can automate the process of checking data for errors, inconsistencies, and outliers, which can save time and reduce the risk of human error.

3. Integration: this macro can be integrated into the data management and analysis process at both the early stage as well as late stage, allowing for a more seamless and efficient workflow.

4. Reusability: this macro can be reused across different projects and studies, which can save time and resources in the long run.

5. Cost-effective: this macro can be less expensive than specialized software tools and can be adapted to the specific needs of a project or study.

CONCLUSION

The macro discussed in this article is used to cross-check discrepancies between ADaM and SDTM datasets during analysis and reporting processes. Also, it provides a framework for cross-checking the discrepancies between datasets and standard files. The framework has three main modules (data/file reading modules, discrepancy checking module, and report creating module) and contains several helpful sub-macros; with each sub-macro performing a specific task or function. These sub-macros can be called within the main macro, allowing the modular macro to be flexible and customizable. The user would use/combine the sub-macros to achieve their specific discrepancy-checking purpose while making the code more organized and easier to read.

REFERENCES

Chao Su, Shunbing Zhao, Cynthia He, 2018, “An Efficient Tool for Clinical Data Check”, *Proceedings of PharmaSUG 2018*, Paper AD-16, Seattle, WA.

Niraj J. Pandya, Vinodh Paida, 2011, “Data Edit-checks Integration using ODS Tagset”, *Proceedings of* PharmaSUG 2011, Paper DM03, Nashville, TN.

CONTACT INFORMATION

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

Chao Su,	Changhong Shi,	Jaime, Yan
+1 (732) 5946459	+1 (732) 5941383	
chao.su@merck.com	changhong_shi@merck.com	mingyu.yan1@merck.com

Any brand and product names are trademarks of their respective companies.

APPENDIX

Sample code used at macros:

```
*-------------------- Macros used at excel output --------------------;

 %macro data2report(indsn =, width =);
    %local er ror i title sheetnam;
    %let er = ER;
    %let ror = ROR;
    %let title = Data Check Results;
    %let sheetnam = &indsn;

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

      *--------- Derive Title and Sheetname used at tab display --------;
    %if &nobs >0 %then %do;
       %if %VarExist(&indsn, title) and %VarExist(&indsn, sheetnam) %then %do;
           proc sql noprint;
             table attr as select distinct title, sheetnam from &indsn;
               select title into: title from attr;
               select sheetnam into: sheetnam from attr
            quit;
       %end;
    %end;

        proc contents data = &indsn
          %if %VarExist(&indsn, title) and %VarExist(&indsn, sheetnam) %then %do;
              (drop = title sheetnam)
          %end;
             %else %if %VarExist(&indsn, title) %then %do;
                 (drop = title)
             %end;
                %else %if %VarExist(&indsn, sheetnam) %then %do;
                    (drop = sheetnam)
             %end;
           out = __indsn_contents(keep =name type length label varnum) varnum noprint;
        run;

        *--------- Derive variable maximum length used at output cell width --------;
       data _null_;
          set __indsn_contents(where = (type = 2)) end=last;
            if _n_ eq 1 then call execute('proc sql noprint;
                                           create table temp as select ');
                                call execute(cat('max(length(',name,')) as ',name ));
            if last then        call execute("from  &indsn ; quit;");
               else             call execute(',');
        run;
        proc transpose data=temp out=_vmaxlen name=name prefix=length;
        run;

        *--------- Derive Cell Width --------;
       proc sql noprint;
         create table _indsn_contents as
            select a.*, b.length1 from __indsn_contents a
              left join
              _vmaxlen b
                 on a.name = b.name;
        quit;

        data _indsn_contents;
            set _indsn_contents;
            name = upcase(name);
              if length1 ne .         then length = length1;
              if length > 10          then length = length;
                 else if length <=10 then length = 10;
        run;
        proc sort data = _indsn_contents;
             by varnum;
        run;

        proc sql noprint;
            %if %length(&width) = 0 %then %do;
                select length into :_columnwidth separated by ','
                from _indsn_contents;
            %end;
            %else %do;
                %let _columnwidth = &width;
            %end;

            select upcase(name) into :_columnvar separated  by ' '
            from _indsn_contents;
        quit;

        title1 j=c bold h=12pt f='Thorndale AMT'  "&title";

        ods excel    options (sheet_name    = "&sheetnam"
                     embedded_titles        = 'yes'
                     autofilter             = 'all'
                     absolute_column_width = "&_columnwidth"
                     zoom                   = '100'
                     orientation            = 'landscape'
                     row_repeat             = 'header'
                     pages_fitheight        = '100'
                     center_horizontal      = 'yes'
                     center_vertical        = 'no'
                     gridlines              = 'on'
                     frozen_headers         = 'yes'
                     start_at               = 'A1')
                     ;


             %if %length(&_columnvar) > 0 %then %do;
                 %do i=1 %to %length(&_columnvar);
                     %if %scan(&_columnvar, &i) ne %str() %then %do;
                         %let _columnvartot=&i;
                     %end;
                 %end;
                 %do i=1 %to &_columnvartot;
                     %let _columnvar&i = %scan(&_columnvar, &i);
                 %end;
             %end;

             options nobyline nolabel;

             %if %length(&_columnvar) > 0 %then %do;
                 %do i=1 %to %length(&_columnvar);
                     %if %scan(&_columnvar, &i) ne %str() %then %do;
                         %let _columnvartot=&i;
                     %end;
                 %end;
                 %do i=1 %to &_columnvartot;
                     %let _columnvar&i = %scan(&_columnvar, &i);
                 %end;
```

```
            %end;

  proc report data = &indsn nofs
     style(header)={font_weight=bold font_size=10pt just=center
                    protectspecialchars=off borderstyle=solid bordercolor=black}
     style(column)={borderstyle=solid bordercolor=black};
     column &_columnvar;
     %do i=1 %to &_columnvartot;
        define &&_columnvar&i /display style(column)={tagattr='wraptext:no'
                                width=100%};  *** Avoid unexpected wrap at display;
     %end;
  run;
%mend data2report;
```