

Multiple Studies! Data DefinitionTracker, Made Easy!

Saritha Bathi, inVentiv Health;

ABSTRACT

Data definition tracking is already a well known challenge in a single study. The complexity of the challenge increases exponentially in the context of large team tasked with handling multiple projects. Our approach to handle the ever changing data definitions effectively comprises of dynamically capturing changes to definitions and make them available to developers at predefined / schedules timeframes. The data definitions are written in a generic excel template and exported into a SAS dataset as a first step. The latest load specification file is compared with the preceding version and a 'delta' file is generated and communicated to the team.

INTRODUCTION

This paper assumes that the data definitions are in excel spread sheet. In general at the start of the study programming, companies start with an initial version of the data definitions spread sheet and keep updating it.

The data files used in the paper are for testing purpose only. The data definitions and names do not represent any factual meanings.

This paper will talk about developing a tracking system that identifies these changes and notifies the changes to the team of programmers and reviewers. This tracker system can be developed using different tools but this paper talks about how SAS is used to develop the system.

Throughout this paper we will talk about the steps involved in the development of the data definition tracking system. The following steps are involved in the development of the tracking system:

1. Setup the environment.
2. Read the initial data definitions excel file to create master metadata.
3. Read the modified data definitions excel file to create changed metadata.
4. Compare the master metadata and the changed metadata.
5. Create the track changes metadata.
6. Generate the track changes report.
7. Notify the changes to the reviewers and programmers.
8. Update the master metadata with the track changes metadata.

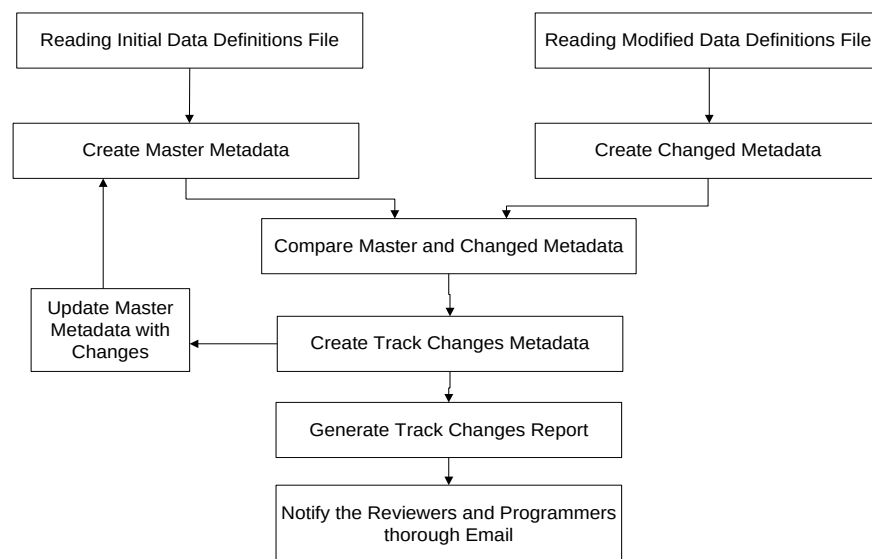


Figure 1. Program Flow

STEP1: SETUP THE ENVIRONMENT

Create and populate the required macro variables to facilitate the development of the Data Definition Tracker application.

```
%global __studyloc__ __ddfile__ __outloc__ __metaloc__ __temploc__ __trdsfile__
__trvarfile__;

%let __studyloc__ = C:\study;
%let __ddfile__ = DataDefinitions_V1.xls;
%let __outloc__ = C:\study\reports;
%let __metaloc__ = C:\study\metadata;
%let __temploc__ = C:\study\temp;
%let __trdsfile__ = DatasetLevelChanges.xls;
%let __trvarfile__ = VarLevelChanges.xls;
```

STEP 2: READ THE INITIAL DATA DEFINITION EXCEL FILE

The data definition spread sheet contains multiple sheets, one sheet for each dataset respectively. In this step read the spread sheet containing the initial data definitions into SAS and create datasets.

NAME	LABEL	TYPE	FORMAT	DERIVATION	VALUES
AGEGRP	Age Group	Char	10	User input. Option to group age [ADSL.BAGE] as defined by user.	
AGE65	Age Group 65	Char	10	Age grouping (two categories only) of [ADSL.BAGE]. Categories are "<65" or ">=65".	
IBRTHDT	Imputed Birth Date	Date	8	Set to date part of ISO Birth Date [DM.BRTHDT] converted to numeric date. Missing Data Handling Rule: - If the day part of the birth date is missing, impute it with 15. - If the month part of the birth date is missing, impute it with June.	
IBRTHDTF	Imputed Birth Date Flag	Char	3	Set to MD if month and day of imputed birth date [ADSL.IBRTHDT] is imputed. Set to Y if day part of imputed birth date is imputed. Set to null if imputed birth date is not imputed.	D MD null
BAGE	Baseline Age	Num	8	Default Option 1: Patient's age (years) at randomization. Set to integer part of (Randomization Date [ADSL.RANDDT] - Birth date [ADSL.IBRTHDT] + 1)/365.25). Option 2: Patient's age (years) at initiation of study treatment. Set to integer part of (First Treatment Date [ADSL.TRTSDT] - Birth Date [ADSL.IBRTHDT] + 1)/365.25). Option 3: Patient's age (mo) at randomization. Set to integer part of (Randomization Date [ADSL.RANDDT] - Birth date [ADSL.IBRTHDT] + 1)/30.4375). Option 4: Patient's age (mo) at initiation of study treatment. Set to integer part of (First Treatment Date [ADSL.TRTSDT] - Birth Date [ADSL.IBRTHDT] + 1)/30.4375). Option 5: Patient's age (yr) at enrollment. Set to integer part of (Inform consent date [ADSL.INFCOIT] - Birth Date [ADSL.IBRTHDT] + 1)/365.25). - For non-randomized studies, if the patient is not treated, replace the first treatment date [ADSL.TRTSDT] with informed consent date [ADSL.INFCOIT] in the derivation.	
BAGEU	Baseline Age Unit	Char	6	Set to 'YEARS' if [ADSL.BAGE] is computed in years. Set to 'MONTHS' if [ADSL.BAGE] is computed in months.	MONTHS YEARS
				Set to baseline Weight value [VS.VSSTRESN] where parameter test [VS.VSSTESTCD] is equal to 'WEIGHT' and baseline flag [VS.VSBLFL] = 'Y'. Note: baseline weight may also be derived in AVS. Dependent on the BLTYPE ('Baseline type') option selected in AVS, the baseline weight derived in AVS may not be identical to that in ADSL.	

Figure 2. Initial Data File

As per the above data definitions file, it has 7 sheets respectively for the 7 datasets.

There are different ways to read excel files in SAS. Sample code below shows how to read in all the worksheets in file DataDefinitions_V1.xlsx and store them in SAS datasets with respect to their sheet names using EXCEL libname engine in SAS version 9.

```

*** libname statement to define libref excellib ***;
*** word excel is the engine name ***;

libname exlib excel "&__studyloc__\&__ddfile__";

%macro readxls (lib=,loc=,ddfl=);

*** Once the library is established, the worksheet ***;
*** names become values of variable memname in the ***;
*** SASHELP dictionary view VSTABVW ***;

*** Create sheetname dataset with all the sheetnames ***;

proc sql noprint;
  create table sheetname as
  select memname as sheetname
  from sashelp.vstabvw
  where libname="EXLIB";

  *** Get the count of the sheet names ***;
  select count(DISTINCT sheetname) into :cnt_sht from sheetname;

  *** Get the sheetnames into macro variables sheet1, sheet2.. respectively ***;
  select DISTINCT sheetname into :sheet1 - :sheet%left(&cnt_sht) from sheetname;
quit;

*** clear the libname to free up the Excel file for future use ***;
libname excellib clear;

libname &lib "&loc";

*** Create all the sheets as datasets with sheetname ***;
%do i=1 %to &cnt_sht;
  proc import datafile="&__studyloc__.\&ddfl"
    out=&&sheet&i replace;
    sheet="&&sheet&i";
    getnames=yes;
    mixed=yes;

  run;
%end;
%mend readxls;
%readxls(lib=meta,loc=%str(&__metaloc__),ddfl=%str(&__ddfile__));

```

STEP 2: READ THE MODIFIED DATA DEFINITIONS EXCEL FILE

Read the modified data definitions file DataDefinitions_V2.xls into SAS and create all the sheets into SAS datasets with respect to their sheet names into the temporary location [can be created in work library]. As per the modified data definitions file AGEGRP variable is deleted, AGE40 variable is added and AGE65 variable is modified. At the dataset level dataset DS7 is deleted and new dataset DS8 is added.

```
%let __ddfile__ = DataDefinitions_V2.xls;

%readxls(lib=temp,loc=%str(&__temploc__),ddfl=%str(&__ddfile__));
```

	A	B	C	D	E	F
	NAME	LABEL	TYPE	FORMAT	DERIVATION	VALUES
2	AGE65	Age Group 65	Char	10	Age grouping (two categories only) of [ASL BAGE]. Categories are "<65" or ">=65"	<65 >=65
3	AGE40	Age Group 40	Char	10	Age grouping (two categories only) of [ASL BAGE]. Categories are "<40" or ">=40"	
4	IBRTHDT	Imputed Birth Date	Date	8	Set to date part of ISO Birth Date [DM.BRTHDT] converted to numeric date. Missing Data Handling Rule: - If the day part of the birth date is missing, impute it with 15. - If the month part of the birth date is missing, impute it with June.	
5	IBRTHDTF	Imputed Birth Date Flag	Char	3	Set to MD if month and day of imputed birth date [ASL IBRTHDT] is imputed. Set to D if day part of imputed birth date is imputed. Set to null if imputed birth date is not imputed.	D MD null
6	BAGE	Baseline Age	Num	8	Default Option 1: Patient's age (years) at randomization. Set to integer part of (Randomization Date [ADSL RANDOIT] - Birth date [ADSL IBRTHDT] + 1) / 365.25. Option 2: Patient's age (years) at the initiation of study treatment. Set to integer part of (First Treatment Date [ADSL TRTSOT] - Birth Date [ADSL IBRTHDT] + 1) / 365.25. Option 3: Patient's age (mo) at randomization. Set to integer part of (Randomization Date [ADSL RANDOIT] - Birth date [ADSL IBRTHDT] + 1) / 30.43735. Option 4: Patient's age (mo) at initiation of study treatment. Set to integer part of (First Treatment Date [ADSL TRTSOT] - Birth Date [ADSL IBRTHDT] + 1) / 30.43735. Option 5: Patient's age (yr) at enrollment. Set to integer part of (Informed consent date [ADSL INFCOOT] - Birth Date [ADSL IBRTHDT] + 1) / 365.25. For non-randomized studies, if the patient is not treated, replace the first treatment date [ADSL TRTSOT] with informed consent date [ADSL INFCOOT] in the derivation.	
7	BAGEU	Baseline Age Unit	Char	6	Set to 'YEARS' if [ADSL BAGE] is computed in years. Set to 'MONTHS' if [ADSL BAGE] is computed in months.	MONTHS YEARS

Figure 3. Changed Data File

STEP 3: COMPARE THE MATSER METADATA AND THE CHANGED METADATA

Compare the datasets from meta library with the temp library datasets and create a SAS dataset explaining what happened to the variables. For example if any attribute changes for a particular variable means that variable is modified. If a variable doesn't appear in the dataset from temp library means that particular variable is removed. If a new variable appears in the temp library dataset means that variable is newly added.

Also check if a dataset is deleted or a new dataset is added.

The following sample code shows a way of finding the new and deleted datasets.

```
PROC SQL;

    *** datasets in the meta(initial) directory ***;
    CREATE TABLE meta_ds AS
    SELECT DISTINCT memname FROM sashelp.vcolumn
    WHERE libname eq "META" AND memtype eq "DATA"
    ORDER BY memname;

    *** datasets in the temp(changed) directory ***;
    CREATE TABLE temp_ds AS
    SELECT DISTINCT memname FROM sashelp.vcolumn
    WHERE libname eq "TEMP" AND memtype eq "DATA"
    ORDER BY memname;

QUIT;

**** datasets in meta library and not in temp library i.e. dataset deleted****;

PROC SQL;

CREATE TABLE deleted AS

SELECT memname FROM meta_ds

EXCEPT SELECT memname FROM compare_ds;
```

```

**** datasets in temp library and not in meta library i.e. dataset added****;
CREATE TABLE added AS
SELECT memname FROM temp_ds
EXCEPT SELECT memname FROM meta_ds;
QUIT;
*** loop through the datasets and compare each dataset and create ***;
*** the changes into tr_sheetname sas datasets ***;
%do i=1 %to &cnt_sht;
    data tr_&&sheet&i(drop=chlabel chtype chformat chderivation chvalues);
        merge meta.&&sheet&i(in=meta)
            temp.&&sheet&i(in=temp rename=(label=chlabel type=chtype format=chformat
            derivation=chderivation values=chvalues));
        length action $10 origin $10;
        by name;
            if meta and temp then do; *** variable exists in both ***;
                if (label ne chlabel) or (type ne chtype) or (format ne chformat)
or (derivation ne chderivation) or (values ne chvalues) then do;
                    action="Modified";
                    origin="Old";
                    output;
                    origin="New:";
                    label=chlabel;
                    type=chtype;
                    format=chformat;
                    derivation=chderivation;
                    values=chvalues;
                    output;
                end;
            end;
            if meta and ^temp then do; *** variable is deleted ***;
                action="Deleted";
                origin="Old:";
                output;
            end;
            if temp and ^meta then do; *** new variable added ***;
                action="Added";
                origin="New";
                label=chlabel;
                type=chtype;
                format=chformat;
                derivation=chderivation;
                values=chvalues;

```

```

        output;
    end;

end;

run;

%end;

```

STEP 4: CREATE THE TRACK CHANGES METADATA

Consolidate all the changes occurred. Create a dataset to report the modifications at the dataset level (added and deleted datasets).

```

data dataset_changes;
    merge added(in=add)
           deleted(in=del); *** add code to check whether the datasets exist***;
    by name;
    length action $20;
    if add then action = "New dataset added";
    if del then action = "Dataset deleted";

run;

```

STEP 5: GENERATE THE TRACK CHANGES REPORT

Create Excel reports with the changes.

The following sample code creates an Excel file with the dataset level changes.

```

PROC EXPORT DATA = dataset_changes
    DBMS=excel OUTFILE = "&__outloc__\&__trdsfile__"
    REPLACE; sheet='dataset_changes';

RUN;

```

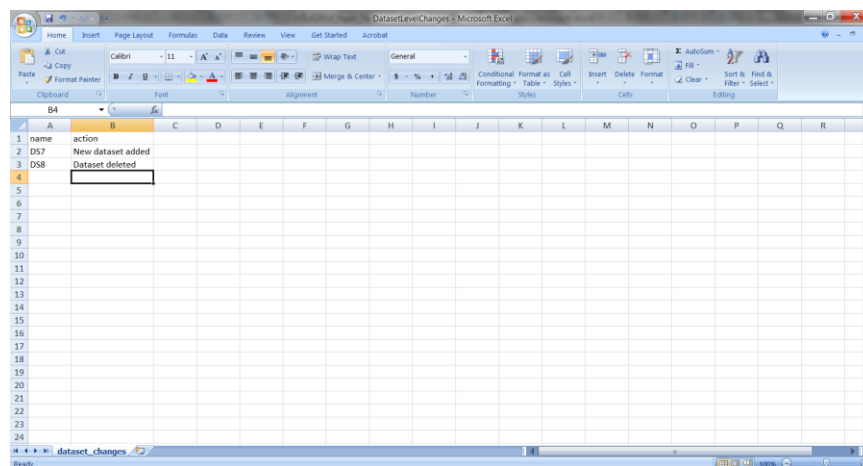


Figure 4. Dataset Level Changes File

The following sample code creates an Excel file with the variable level changes.

```

*** Sheet names will be dataset names ***;
%do i=1 %to &cnt_sht;
    PROC EXPORT DATA = "tr_&&sheet&i";
        DBMS=excel OUTFILE = "&__outloc__\&__trvarfile__"
        REPLACE; sheet = "&&sheet&i";
    %end;

```

```
RUN; %end;
```

	A	B	C	D	E	F	G	H	I	J	K	L
1	name	action	origin	label	type	format	derivation	values				
2	AGEGRP	Deleted	Old:	Age Group 65	Char	10	Age grouping (two categories only) of [ASL.BAGE]. Categories are "<65" or ">=65"					
3	AGE40	Added	New:	Age Group 40	Char	10	Age grouping (two categories only) of [ASL.BAGE]. Categories are "<40" or ">=40"					
4	AGE65	Modified	Old:	Age Group 65	Char	10	Age grouping (two categories only) of [ASL.BAGE]. Categories are "<65" or ">=65"					
5			New:	Age Group 65	Char	10	Age grouping (two categories only) of [ASL.BAGE]. Categories are "<65" or ">=65"	<65 >=65				

Figure 4. Variable Level Changes File

STEP 6: NOTIFY THE CHANGES TO THE REVIEWERS AND PROGRAMMERS

Send an email to the team with the changes in the derivations as clear attachments.

```
FILENAME outbox EMAIL ("Saritha.Aleti@inventivhealth.com");

DATA _NULL_;

FILE outbox TO=("first.programmer@inventivhealth.com" "first.reviewer@
inventivhealth.com")

FROM=("Saritha.Aleti@inventivhealth.com ")

SUBJECT=("Changes in the derivations of STUDY XXX" )
ATTACH=("&__outloc__\&__dsvarfile__" "&__outloc__\&__trvarfile__" );

PUT " ";

PUT "Dear All,";

PUT " ";

PUT "Attached are the Data Definitions Track Changes Reports.";

PUT " ";

PUT "Saritha Aleti";

RUN;
```

STEP 7: UPDATE THE MASTER METADATA WITH THE TRACK CHANGES METADATA

Clear the meta library and update the meta library from the temp library.

```
*** delete all datasets from meta (initial) library ***;
proc datasets library=meta kill;run;
quit;

*** Update the meta library with temp(changed) datasets which ***;
*** becomes base for the upcoming changes ***;
proc datasets nolist;
copy in=temp out=meta; run; quit;
```

CONCLUSION

SAS is a powerful tool that provides programmers with a wide range of solutions and options. This paper may not be taking advantage of the full capabilities of SAS. This paper talks more about the idea of the process. Some of the techniques used in the paper can be changed and replaced by more efficient methods. The Data Definitions Tracker is very useful, once developed can be used in all study teams irrespective of the therapeutic areas.

ACKNOWLEDGMENTS

I would like to thank inVentiv Health Clinical, for giving me the opportunity to present this paper.

CONTACT INFORMATION

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

Saritha Bathi [Aleti]
inVentiv Health Clinical
10 Forest Dr
Piscataway, NJ 08854
(732) 986 - 4180
Saritha.aleti@inventivhealth.com

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

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