PhUSE 2011

Paper DH07

Much ADaM about Nothing – a PROC Away in a Day

Endri, i3, Berlin, Germany
Rowland Hale, i3, Berlin, Germany

ABSTRACT

CDISC is rapidly becoming adopted as the data standard for clinical trials and submissions of clinical trial data to the FDA. Within CDISC, ADaM datasets are an integral part of clinical study analysis and require significant data derivation to fulfill the needs of TLF provision. Can the creation of ADaM datasets be automated? Yes! This paper proposes an Excel driven solution which greatly improves the efficiency and accuracy of ADaM dataset creation by defining ADaM dataset structure, complex variable derivation and data checks within Excel. Using a library of SAS macros, the definitions then drive the automated production of SAS scripts which produce analysis ready datasets that meet the ADaM specification and which can be validated in accordance with regulatory requirements. Additionally, data check reports are created for data management to speed up the data cleaning process.

INTRODUCTION

CDISC ADaM version 2.1 is now released on the CDSIC website. This document describes the fundamental principles of the ADaM analysis dataset and the design and purpose of submitted analysis datasets. The main principle of ADaM datasets is “analysis-ready” so that only minimal programming effort is needed to achieve the statistical results. More than one single statistical procedure may be required to calculate the statistical outputs, but the goal of ADaM is minimum programming effort and maximum concentration on the results.

The ADaM Implementation Guide v.1.0, which is posted on the CDISC Website, describes in detail the standard structure of ADaM datasets, the variables they contain and the derivation methods used.

Using metadata to define the structure and derivation methods of ADaM datasets greatly facilitates the automation of ADaM dataset creation directly from CDISC SDTM datasets. This paper outlines the fundamentals of our approach and describes a step by step ADaM dataset automation process.

METHODS

ADAM DATASETS – PRINCIPLES OF DERIVATION

The following diagram shows the flow of standard clinical trial data.

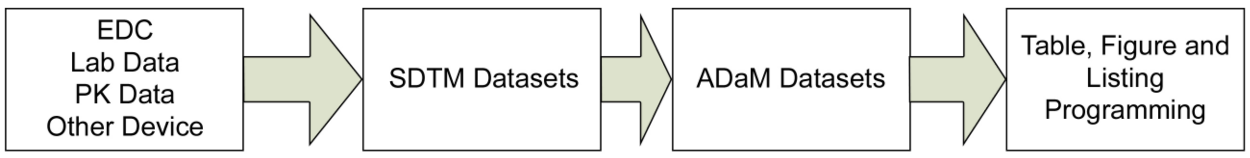

Figure 1: Flow of standard clinical data

As CDISC SDTM – which contains only a small number of derived variables – becomes the standard structure for clinical trial data, ADaM datasets contain all derived variables required for the analysis process. This means that ADaM datasets should aim to be analysis ready or “one proc away” for the statistical outputs.

The analysis variables derived for one particular ADaM Dataset, e.g. ADEG, might be different from those used for another dataset, e.g. ADVS, and all of these should be clearly documented and linked in the metadata to ensure traceability. Besides information about the content, structure and source of the data, the metadata also indicate the derivation or imputation method of the analysis variable.

The ADaM Implementation Guide describes two standard data structures:

1. Subject-Level Analysis Dataset (ADSL Dataset)
   This dataset is the minimum requirement for ADaM. It provides key information for each subject in the clinical trial and has a single record per subject in the dataset.
2. Basic Data Structure (BDS Dataset)
   The primary keys of the BDS dataset are subject, analysis parameter and (dependant upon the analysis) analysis timepoint.
   Although some variables are required to exist within an ADaM dataset, e.g. AVAL, PARAM, DTYPE, etc., the BDS is flexible in terms of additional rows and columns. This allows BDS datasets to provide robust and flexible support for most types of derivation and statistical analysis.
   Further information about these ADaM variables can be found in the ADaM Implementation Guide v.1.0.

Understanding the structure of CDISC SDTM Datasets and the ADaM Implementation Guide helps us in automating the derivation of ADaM Datasets. In general, the following steps are required to create ADaM datasets:

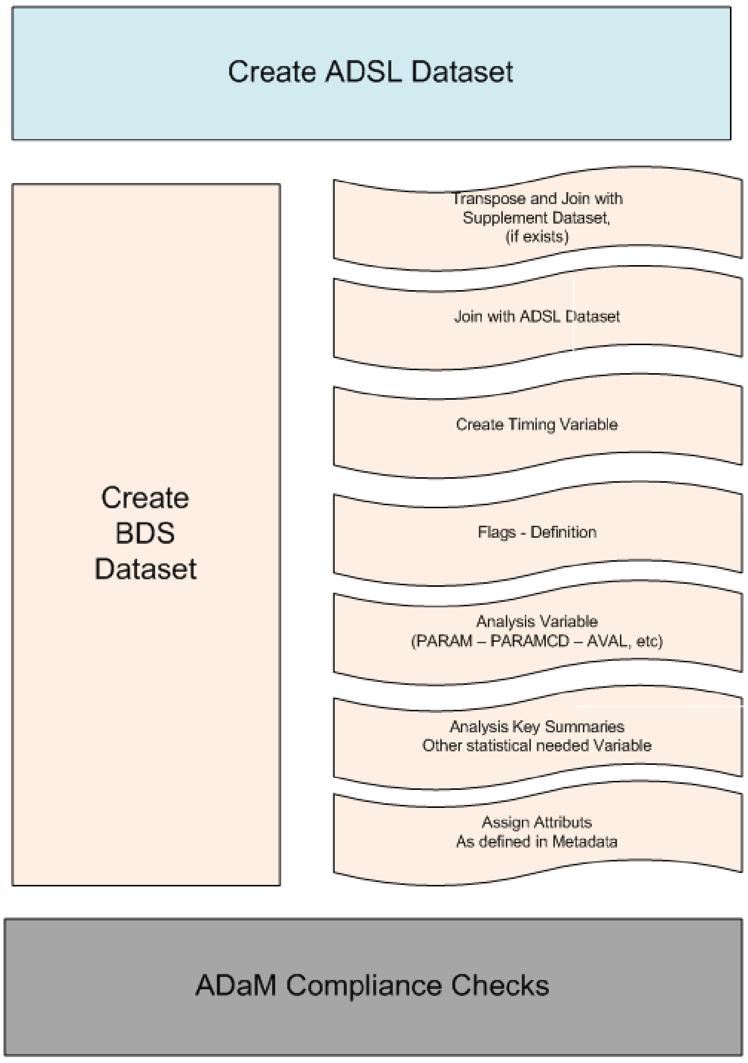
Create ADSL Dataset
Create BDS Dataset
Transpose and Join with Supplement Dataset, (if exists)
Join with ADSL Dataset
Create Timing Variable
Flags - Definition
Analysis Variable (PARAM – PARAMCD – AVAL, etc)
Analysis Key Summaries Other statistical needed Variable
Assign Attributs As defined in Metadata
ADaM Compliance Checks

Figure 2: Create ADaM Datasets

ADAM METADATA AND DERIVATION METHODS FOR THE AUTOMATION PROCESS

As described above all variables for each ADaM dataset, their attributes and their derivation method are specified in the metadata. For clarity we define each analysis dataset in a separate Excel worksheet. This makes it easier to add new variables to a particular dataset if needed.

Keeping the metadata as simple as possible without reducing flexibility is the key to efficient work and the successful implementation of the automation process:

The analysis dataset metadata has the following configuration and key variables:

- Spreadsheet Name	: used to identify the analysis dataset name.
- Variable Name	: variable name
- Variable Label	: variable label
- Type	: variable type (NUM or CHAR)
- Length	: variable length (e.g. for NUM the length may be 8)
- Format	: variable format
- Source Table	: library and the table name of the source data
- Source Variable	: source variable (e.g. EGSTRESN from the EG domain in SDTM is the source for AVAL in the ADEG ADaM dataset)
- Derivation Method	: describes the derivation method of the particular variable.

Where variables in the ADaM dataset have more than one source variable, we can use a special character e.g. # or § to delimit the list of source variables.

The metadata may also contain variables required for basic and special data checks such as primary keys, not null or others. Such data checks do not normally fall within the statistical programming remit, yet they are quick and easy to implement within the system and can help us to identify data issues in the statistical analysis outputs. Basic data checks are described later.

Examples of analysis metadata for the automation process are shown below.

Sort	Meta	Name	Label	Type	Length	Format	SourceTab	SourceVar	DerivedMeth
100	Table	ADSL	Subject Level Analysis Data				SDTM.DM		
200	Var	STUDYID	Study Identifier	char	6				
300	Var	USUBJID	Unique Subject Identifier	char	15				
400	Var	SITEID	Study Side Identifier	char	15				
500	Var	AGE	Age in AGEU at RFSTDTC	num	8				
600	Var	WEIGHT	Weight (kg)	num	8		SDTM.VS	VSSTRESN	Q: WHERE vstestcd = 'WEIGHT' and visit = 'SCREENING'
700	Var	HEIGHT	Height (cm)	num	8		SDTM.VS	VSSTRESN	Q: WHERE vstestcd = 'HEIGHT' and visit = 'SCREENING'
800	Var	TR01SDT	Date of First Exposure in Period 01	num	8	DATE9	SDTM.EX	EXSTDTC	#FirstDate

Table 1: ADSL Metadata (excerpt)

Sort	Meta	Name	Label	Type	Length	Format	Informat	SourceTab	SourceVar	DerivedMeth
100	Table	ADEG	Analysis Dataset for Electrocardiogram					SDTM.EG		
200	Var	STUDYID	Study Identifier	char	6					
300	Var	DOMAIN	Domain Abbreviation	char	2					
400	Var	USUBJID	Unique Subject Identifier	char	15					
500	Var	EGSEQ	Sequence Number	num	8					
600	Var	EGTESTCD	ECG Test or Examination Short Name	char	8					
700	Var	EGTEST	ECG Test or Examination Name	char	40					
800	Var	EGCAT	Category for ECG	char	30					
2100	Var	EGDTC	Date/Time of ECG	char	20					
2200	Var	EGENDTC	End Date/Time of ECG	char	20					
2300	Var	EGTPTNUM	Planned Time Point Number	num	8					
2400	Var	EGTPT	Planned Time Point Name	char	50					
2500	Var	AVAL	Analysis Value	num	8			SDTM.EG	EGSTRESN	
2600	Var	CHG	Change from Baseline	num	8			DERIVED		AVAL - BASE
2700	Var	BASE	Baseline Value	num	8			SDTM.EG	EGSTRESN	Q: WHERE EGBLFL = 'Y'
2800	Var	AVAL2	Analysis Value - Converted from EGSTRE	num	8			SDTM.EG	EGSTRESC	#Char2Num
2900	VAR	CHG1G	Change Group	char	8			DERIVED		IF (CHG/BASE) < 10 THEN chg1g = 1; ELSE chg1g=2;

Table 2: ADEG Metadata (excerpt)

A library of standard macros is used for common derivation requirements. More complex or study-specific derivations which cannot be achieved through use of the standard macros are handled by ad hoc SAS plug-ins incorporated into the output scripts.

These standard derivation macros, which are also listed in the Excel spreadsheet, will be automatically applied to the mapping scripts during the automation process.

An example of library of standard macros for the automation process is shown below.

Name	Description	LibMacro	PreProcessing	Processing	PostProcessing
MinDate		`%mindate( intab = ##INTAB## , outtab = ##OUTTAB## , outdate = ##NAME## , date = ##AUTO##);`			
Char2Num		`%_help_c2n(var = ##AUTO##)`			
AutoJoin		`%auto_join ( intab = ##INTAB## , outtab = ##OUTTAB##);`			
HelpSort				`PROC SORT DATA = ##INTAB## OUT = ##OUTTAB##; BY USUBJID ##BY##; RUN;`	
FirstDate			#HelpSort	#MinDate	

Table 3: Example of Macro Libraries

There are three different kinds of macros, for flexibility in the automation process. These are:

1. Macros starting with "_"
   These macros have the meaning of a "submacro" that does nothing but derive a variable within the datastep or SQL procedure, e.g.
   #Char2Num → %_help_c2n(var = ##AUTO##);
2. Macros without the leading "_"
   After applying these macros we will have a new dataset, e.g.
   ```
   #AutoJoin → %auto_join ( intab   = ##INTAB##
                          , outtab  = ##OUTTAB## );
   ```
3. Macros enabling pre- and post-processing
   These macros enable pre- and post-processing to solve certain difficult programming problems.

AUTOMATION PROCESS

User friendly and easily read by SAS, Excel provides a convenient format for storing metadata to drive automated systems. Dataset structure, derivation method and data checks are all defined within an Excel "driver sheet" from which a set of SAS macros generates the SAS scripts that, in turn, create the ADaM datasets.

General steps during the automation process:

1. Import the ADaM metadata into SAS.
2. Import the functional core (Macro Libraries) into SAS.
3. Attach the functional core definition to the ADaM Metadata, if any exist
4. The "Definition Step"
5. Creating mapping scripts based on the metadata.

For example, the metadata of the ADEG dataset define four derived variables (AVAL, CHG, BASE and CHG1G). We know from the metadata that we need to derive AVAL and BASE first before deriving the CHG and CHG1G which are dependent on them. This small example demonstrates the importance of the "Definition Step" in the automation process.

Accordingly, the "Definition Step" needs to define the order of derivation, independently of how these variables are sorted in the metadata. The simple solution to this definition step is to loop over each observation, performing a keyword search of the variable "DerivedMeth" in the metadata and the Processing Variable in the function core.

Another important case within this "Definition Step" is to identify derived variables on which other derived variables depend. In our case, the BASE variable is defined separately as the baseline value, without retaining, and must be merged back onto the dataset.

Two further aspects of good programming practice for clinical trials are 1) the insertion of parentheses in meaningful places in order to clarify the order of precedence of mathematical or logical operations and 2) the automatic insertion of comments into the mapping scripts.

The macro call for this whole process is:

```
   %ADaM_Gen ( metadata      = ADEG
             , runall        = N
             , addobs        = ENDPOINT # AVERAGE
   );
```

The macro `%ADaM_Gen` generates the ADEG mapping script which produces all variables defined in the metadata and creates derived observations with the calculated value of DTYPE = ‘ENDPOINT’ and ‘AVERAGE’.

```
/***********************************************************
* Program name      : ADEG_script.sas
* (This script is automatic generated by %ADaM_Gen
* Author            : Endri
* Date created      : 29.07.2011
* Study             : (Study number)
*                     (Study title)
* Purpose           : ADEG - Analysis Dataset for Electrocardiogram
* Template          :
* Inputs            :
* Outputs           :
* Program completed : Yes/No
* Updated by        : (Name) - (Date):
*                            (Modification and Reason)
**********************************************************/

/* Analysis Value # Analysis Value - Converted from EGSTRESC - SDTM.EG */
DATA adeg_010_eg;
  SET sdtm.eg;
  aval = egstresn;
  aval2 = %_help_c2n(var = egstresc);
RUN;

/* Baseline Value - SDTM.EG */
DATA adeg_020_base;
  SET sdtm.eg;
  base = egstresn;
  WHERE EGBLFL = 'Y';
RUN;

/* Join all */
%auto_join ( intab  =   adeg_010_eg
                      $ adeg_020_base # base
           , outtab = adeg_30_all);

/* Change from Baseline # Change Group - DERIVED */
DATA adeg_40_chg;
  SET adeg_30_all;
  chg = AVAL - BASE;
  IF (CHG/BASE) < 10 THEN chg1g = 1;
                     ELSE chg1g = 2;
RUN;

/* Adding derived observation - DERIVED*/
%calc( intab  = adeg_40_chg
     , outtab = adeg_50_calc
     , calc   = ENDPOINT # AVERAGE);

/* Assign label and data check */
%assign_attrib( metadata = ADEG
              , intab    = adeg_50_calc);
```

Example 1: ADEG - SAS Script (excerpt written by the automation macro)

```
/*********************************************************
* Program name      : ADSL_script.sas
* (This script is automatic generated by %ADaM_Gen
* Author            : Endri
* Date created      : 29.07.2011
* Study             : (Study number)
*                     (Study title)
* Purpose           : ADSL - Subject Level Analysis Data
* Template          :
* Inputs            :
* Outputs           :
* Program completed : Yes/No
* Updated by        : (Name) - (Date):
*                            (Modification and Reason)
*********************************************************/

/* SDTM.DM */
DATA adsl_010_dm;
  SET sdtm.dm (KEEP = studyid usubjid siteid age);
RUN;

/* Weight (kg) - SDTM.VS */
DATA adsl_020_weight;
  SET sdtm.vs;
  weight = vsstresn;
  WHERE vstestcd = 'WEIGHT' and visit = 'SCREENING';
RUN;

/* Height (cm) - SDTM.VS */
DATA adsl_030_height;
  SET sdtm.vs;
  height = vsstresn;
  WHERE vstestcd = 'HEIGHT' and visit = 'SCREENING';
RUN;

/* Date of First Exposure in Period 01 - SDTM.EX */
PROC SORT
  DATA = sdtm.ex
  OUT  = adsl_040_tr01sdt;
  BY     usubjid exstdtc;
RUN;

%mindate( intab    = adsl_040_tr01sdt
        , outtab   = adsl_041_mindate
        , outdate  = tr01sdt
        , date     = exstdtc);

/* Join all */
%auto_join ( intab  =   adsl_010_dm
                      § adsl_020_weight # weight
                      § adsl_030_height # height
                      § adsl_041_mindate # tr01sdt
           , outtab = adsl_50_all);

/* Assign label and data check */
%assign_attrib( metadata = ADSL
              , intab    = adsl_50_all);
```

Example 2: ADSL - SAS Script (excerpt written by the automation macro)

VALIDATION

The system automatically creates ADaM datasets by producing a series of SAS scripts, one per ADaM dataset, which can then be validated against the original ADaM dataset specifications. This ensures that validation can take place in full compliance with SOPs. And because program structure is controlled, consistency across studies and projects is achieved and validation further facilitated.

There are several advantages to a system that produces SAS Scripts automatically. These include easy validation of some scripts, especially certain difficult cases in the derivation process, and also the avoidance of syntax and logical errors. When a problem is solved the solution can be reused without additional validation in a kind of “algorithm template” library.

DATA CHECKS

The Excel driver sheet may include basic data checks such as cross-dataset referencing to ensure completeness of the data and checks to ensure that values are within range or not null. A data error report is generated and this can be passed to data management to facilitate data cleaning. This should cover only a small data quality check such as primary keys, not missing values and even small cross checks with other datasets.

Sort	Meta	Name	Label	ManualCheck
100	Table	ADEG	Analysis Dataset for Electrocardiogram	
200	Var	STUDYID	Study Identifier	
300	Var	DOMAIN	Domain Abbreviation	
400	Var	USUBJID	Unique Subject Identifier	DM.USUBJID
600	Var	EGTESTCD	ECG Test or Examination Short Name	
700	Var	EGTEST	ECG Test or Examination Name	
800	Var	EGCAT	Category for ECG	
2100	Var	EGDTC	Date/Time of ECG	
2200	Var	EGENDTC	End Date/Time of ECG	
2500	Var	AVAL	Analysis Value	
2600	Var	CHG	Change from Baseline	
2700	Var	BASE	Baseline Value	
2800	Var	AVAL2	Analysis Value - Converted from EGSTRESC	
2900	VAR	CHG1G	Change Group	

Table 4: ADEG Metadata - Manual check (excerpt)

Further SAS macros to ensure that the generated ADaM Datasets are compliant with the ADaM Implementation Guide can be built into the automation macro and scheduled to run after the ADaM dataset is created.

CONCLUSION

Long experience of clinical programming and sound knowledge of the SDTM and ADaM Implementation Guides are key to the successful development of a system like this. Once the automation system is created, much less time and effort are needed to fill out the Excel metadata according to the SAP, and manual programming is only needed for difficult cases.

Other advantages to this system are:

- The automatically created mapping script is standard so that double programming is no longer required.
- Documentation of the Analysis Datasets is already done and only needs adjustment if required.
- Basic data checks for Data Management are possible as needed.
- Compliance Checks with the Implementation Guide are also provided at the end of the mapping process.

ACKNOWLEDGMENTS

The authors would like to thank Keith Falconer (Principal Statistical Programmer, i3) for his inputs and his thoughtful review of this manuscript.

REFERENCES

Analysis Data Model (ADaM) Implementation Guide, Version 1.0. (http://www.cdisc.org)
Analysis Data Model (ADaM), Version 2.1. (http://www.cdisc.org)

CONTACT INFORMATION

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

Endri
i3
Knesebeckstr. 30
Berlin, 10623
Germany
Work Phone: +49 (0) 30 345 069 226
Email: Endri.Endri@i3global.com
Web: www.i3global.com

Rowland Hale
i3
Knesebeckstr. 30
Berlin, 10623
Germany
Work Phone: +49 (0) 30 345 069 10
Email: Rowland.Hale@i3global.com
Web: www.i3global.com