Paper S106

How to reduce programming time for ADaM Creation? – Presenting STAG, a Tool Based Approach

Diganta Bose, Cytel Statistical Software & Services P.v.t. L.t.d., Bangalore, India

ABSTRACT

In this Industry, fast track systems are needed to push maximum "electronic submission" deliverables in minimum possible time. Keeping this "objective" in mind, it is always handy to have tools that can process study level analysis results quickly and conveniently with minimal "study specific or data driven" programming intervention. **SDTM To ADaM Generator** Tool, or **STAG**, sets the example of "tool based ADAM automation" that directly intends to contribute towards enhancing productivity of CDISC based submissions to regulatory authorities.

The tool, at front end, has a User Interface that asks the user to fill out certain fields so that study level analysis requirements can be conveniently specified in a simple and succinct manner. Once the application is run, SAS® Program files are generated. These SAS® program files will contain codes that can be utilized to create the most appropriate and important ADaM variables based on the entries in the UI.

INTRODUCTION

The introduction explains the purpose and scope of your paper and provides readers with any general information they need to understand your paper.

STAG (SDTM To ADaM Generator) is the *"first of its kind" attempt to automate ADaM Variable Derivations* in this industry. There are plenty of tools already used for ADaM Standardization within client specific systems however there are no tools available that could automate the "actual ADaM Derivations". ADaM Standardization in this context refers to automation techniques used to ensure compliance to CDISC ADaM Standards and this process is independent of implementing the underlying complex derivations to cater to study specific analysis requirements. Automating ADaM Standardization is easy and only includes check programs that maps a readily available metadata to the output without actually deriving the variables involved. On the other hand, automating the actual ADaM Derivations is highly challenging because Analysis requirements vary from one clinical trial design to another and writing

generalized automation programs is extremely difficult.

ADaM is dependent on the Statistical Analysis Plan and open to interpretation. STAG does not aim to generalize study specific interpretations but rather intends to make customizations based on user requirements. This tool can be used effectively to derive 60-70% of ADaM ADSL and BDS variables, provided SDTM datasets are ready for a given study. The tool, at the front end, has a User Interface (which is designed in a user-friendly manner) that asks the user to fill out certain fields so that study level analysis requirements can be conveniently specified in a simple and succinct manner. These inputs eventually are used in the backend to pass parameters to robustly programmed SAS® Utility macros. Once the fields in the UI are filled out and the application is run, SAS® Program files will be generated. The user will have full control on the program file through the UI Layer. These SAS® program files will contain codes that can be utilized to create the most appropriate and important ADaM variables based on the entries in the UI. These codes can be modified further (if at all very specific additional variables are needed) and run to generate the final ADAM dataset in question.

"**Under normal circumstances**, given the fact that there is no automation tool available for actual ADaM Derivation in the market, ADaM Implementation programming projects for a clinical study usually will involve a programming team of 3-4 Programmers with each programmer writing data driven programming steps for each of the ADaM Outputs. In such a situation an ADaM Dataset of average complexity would take **8-12 hours of programming time to complete.** Each Clinical Study of average complexity would contain an average of 15-20 ADaM outputs.
However, with this tool, the programming steps need not be written by user at all. In fact, based on scenario specific user's inputs, the tool will be writing the programming steps needed to generate the ADaM output in question; the user just needs to run the program file generated by this tool. **Using STAG**, an ADaM dataset of average complexity is just **a few clicks away and should not take more than few minutes to complete**."

WORK FLOW

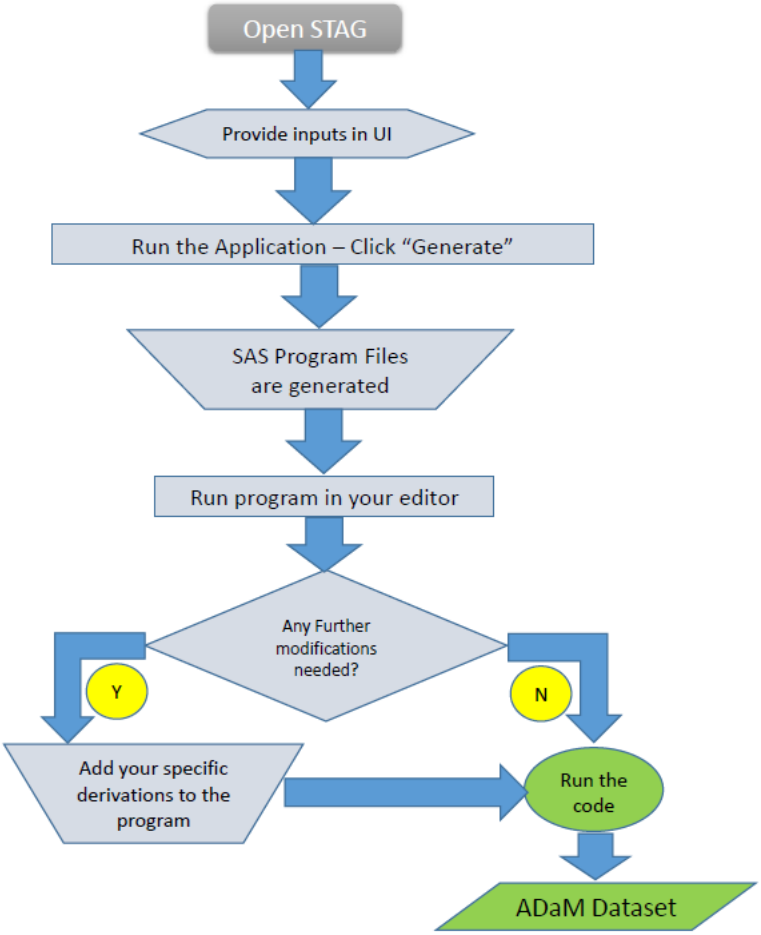
The basic pre-requisite is that the entire SDTM Datasets for a given study should be validated and ready to use, which also includes the Trial Design Datasets like TA, TE and so on. Ideally if these datasets are passed through standard tools like Pinnacle 21 and validated by double programming Quality Control methods, it can be assumed that the SDTM package is ready to use. The reason for this is because, SDTM datasets are the main inputs for creating ADaM Datasets. Additionally, the SAS® Macro programs used backend often search for the datasets it expects in order to create certain variables, for Example, in order to create treatment variables that depend on periods, variables TRTXXA and TRTXXP, the programs backend will need to work with

all SDTM datasets that give information about the Trial Design of the study in question like TE, TA, SE, DM, DS and EX and so on.

The Tool package comprises of few SAS® Program Files, one normal excel file that provides a template to capture Visit Window information that the user needs to fill out and upload in the tool, and one macro enables excel file which contains the front end User Interface of STAG.

The tool needs to be clicked and opened using VBA Excel, and basic prerequisite is SAS® Installation. The UI will have two radio buttons, ADSL and BDS, depending on which the user will choose based on what kind of ADaM needs to be created. Depending on this, the UI entry forms will appear, and basic information needs to be filled in. Below is a flowchart that explains the workflow.

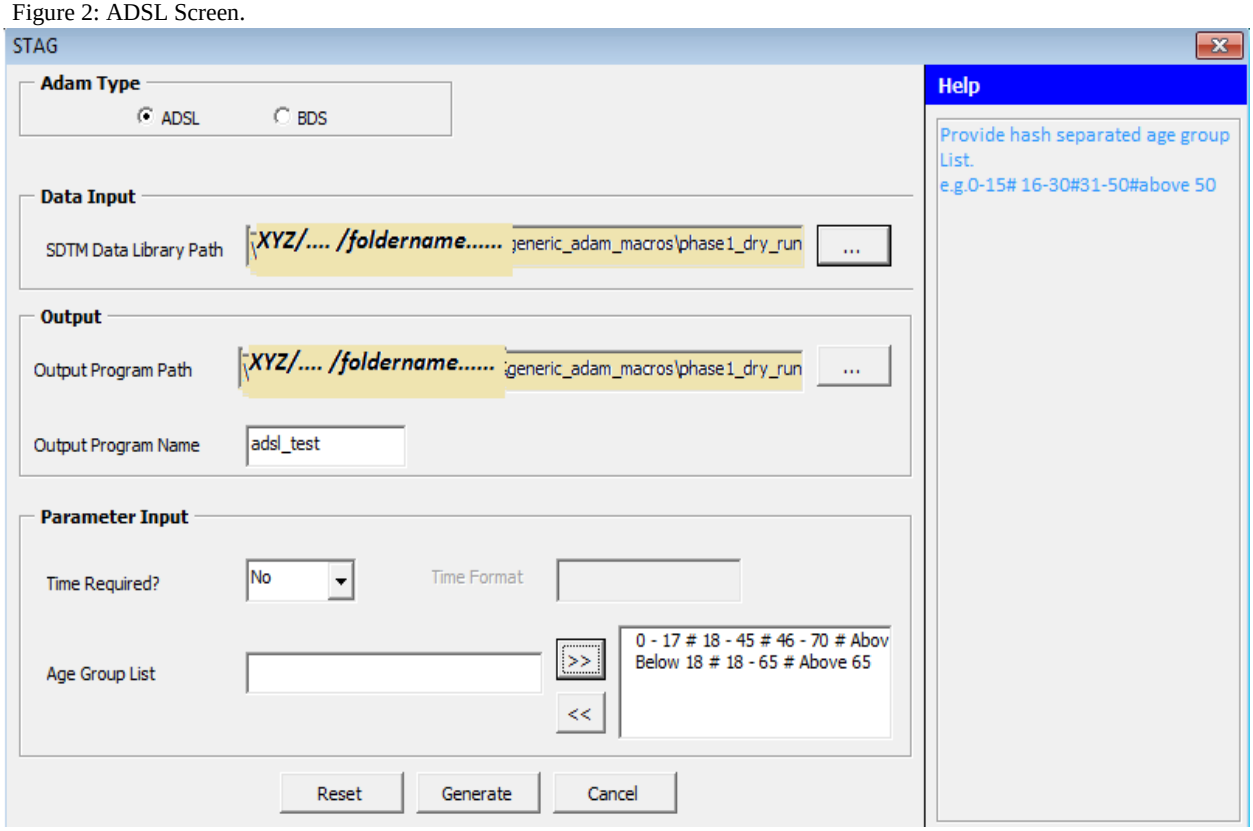
Figure 1: STAG Process Flow



CREATING A STANDARD ADSL

Once the user selects “ADSL” radio button, a screen will appear and it will ask the user to enter some details like location of SDTM Datasets, the location and name of the program file which the user wishes to generate, an option for the user to specify whether timeparts needs to be captured for the treatment related dates, and the specifications for the age group related variables that need to be derived in the ADSL. The details of the ADSL Screen is shown below.

Figure 2: ADSL Screen.



ADSL CREATION EXAMPLE 1

Below shows some typical SDTM Datasets that need to be used to creating basic ADSL. In this example, entries are made in the UI in a similar manner as shown above in *Figure 2: ADSL Screen*, and SDTM Datasets like below exists as shown in *Figure 3: Sample SDTM Inputs*. A SAS® program file “adsl_test.sas” will be generated at the location specified by user once the use clicks the “Generate Button”. The lines of code printed in “adsl_test.sas” is shown in *Figure 4: Sample Code Generated*. As you can see, there are multiple macros backend that are being called and explaining the details of the backend SAS® programming is beyond the scope of this paper. On running the SAS® code, the Sample ADSL dataset that gets generated is shown in *Figure 5: Sample ADSL Dataset*.

Figure 3: Sample SDTM Inputs

DM Data

	STUDYID	DOMAIN	USUBJID	RFSTDTC	RFENDTC	AGE	SEX	RACE	ARMCD	ARM
1	CDISC01	DM	CDISC01.100008	2003-04-29	2003-10-12	72	M	OTHER	WONDER10	Miracle Drug 10 mg
2	CDISC01	DM	CDISC01.100014	2003-10-15	2004-03-29	66	F	WHITE	WONDER20	Miracle Drug 20 mg
3	CDISC01	DM	CDISC01.200001	2003-09-30	2004-02-02	80	F	MULTIPLE	PLACEBO	Placebo
4	CDISC01	DM	CDISC01.200002	2003-10-10	2004-03-28	70	F	BLACK OR AFRI...	WONDER10	Miracle Drug 10 mg
5	CDISC01	DM	CDISC01.200005			66	F	WHITE	SCRNFAIL	Screen Failure

EX Data

	STUDYID	DOMAIN	USUBJID	EXSEQ	EXTRT	EXSTDTC	EXENDTC
1	CDISC01	EX	CDISC01.100008	1	Miracle Drug	2003-04-29	2003-05-05
2	CDISC01	EX	CDISC01.100008	2	Miracle Drug	2003-05-06	2003-05-12
3	CDISC01	EX	CDISC01.100008	3	Miracle Drug	2003-05-13	2003-05-19
4	CDISC01	EX	CDISC01.100008	4	Miracle Drug	2003-05-20	2003-05-27
5	CDISC01	EX	CDISC01.100014	1	Miracle Drug	2003-10-15	2003-10-21
6	CDISC01	EX	CDISC01.100014	2	Miracle Drug	2003-10-22	2003-10-30
7	CDISC01	EX	CDISC01.100014	3	Miracle Drug	2003-10-31	2003-11-11
8	CDISC01	EX	CDISC01.200001	1	Placebo	2003-09-30	2003-10-06
9	CDISC01	EX	CDISC01.200001	2	Placebo	2003-10-07	2003-10-13
10	CDISC01	EX	CDISC01.200001	3	Placebo	2003-10-14	2003-10-20
11	CDISC01	EX	CDISC01.200001	4	Placebo	2003-10-21	2003-10-27
12	CDISC01	EX	CDISC01.200002	1	Miracle Drug	2003-10-10	2003-10-16
13	CDISC01	EX	CDISC01.200002	2	Miracle Drug	2003-10-17	2003-10-18
14	CDISC01	EX	CDISC01.200002	3	Miracle Drug	2003-10-19	2003-10-23
15	CDISC01	EX	CDISC01.200002	4	Miracle Drug	2003-10-24	2003-10-30
16	CDISC01	EX	CDISC01.200002	5	Miracle Drug	2003-10-31	2003-11-05
17	CDISC01	EX	CDISC01.200002	6	Miracle Drug	2003-11-06	2003-11-06

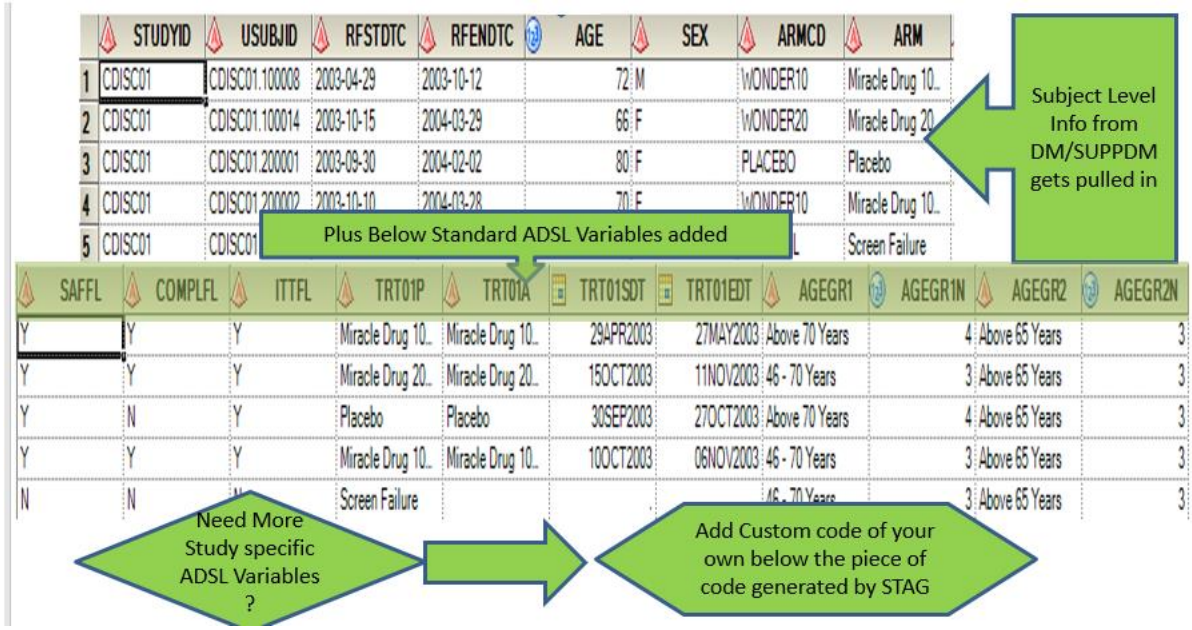
Figure 4: Sample Code Generated

Sample Code Generated By STAG

```
*=============================================================================================

libname sdtm_dt 'XYZ/.... /foldername.....
%include '/XYZ/.... /foldername.....
%include '/XYZ/.... /foldername.....
%include '/XYZ/.... /foldername.....

/*Create dummy_ex with formatted EXTRT to match with ARM and ACTARM*/
%dummy_ex(sdtmlib=sdtm_dt);

/*Create pre_adsl dataset in work library*/
%generic_adsl(sdtmlib=sdtm_dt, exds=dummy_ex, timereq=No, agegroup=0 - 17 # 18 - 45 # 46

/*If more variables need to be derived in pre_adsl write your code below*/
```

Run the piece of code generated by STAG to assess to what extent ADaM ADSL variables are derived

Figure 5: Sample ADSL Dataset (as generated by running the code shown in figure 4.



ADSL CREATION EXAMPLE 2

As we can imagine, there could be multiple study designs that the tool me encounter. The macro programs that work backend are programmed robustly to take into account all kinds of scenarios that may arise due to differing study design aspects. Unlike example 2, this example shows how treatment variables get created by this tool if a study in question has 4 treatment arms in different combinations. In this example, a subject is randomized to receive treatments A, B, C and D in various combination depending on randomized group each subject is assigned to. *Figure 6: Sample input*, shows the SDTM Datasets used to generate then output shown in *Figure 7: Sample Output*. Notice how in this case the variables TRTXXA and TRTXXP are derived based on the scenario, once then code generated by the tool is run. Also note that the treatment dates and TRTXXA are missing for the subject 005, since this subject is not in the safety population and haven't received any actual treatments based on randomization.

Figure 6: Sample input

DM Data

STUDYID	USUBJID	BRTHDTC	SEX	ARM	ARMCD	RFSTDTC	RFENDTC
CDISC01	001	1930-08-05	M	drugA-drugB-dru...	A-B-C-D	2010-01-09	2010-04-14
CDISC01	002	1936-11-01	F	drugD-drugC-dru...	D-C-A-B	2010-01-11	2010-04-15
CDISC01	003	1923-09-03	F	drugB-drugA-dru...	B-A-C-D	2010-01-14	2010-04-20
CDISC01	004	1933-07-22	F	drugA-drugB-dru...	A-B-D-C	2010-01-18	2010-04-22
CDISC01	005	1937-02-22	F	drugB-drugC-dru...	B-C-A-D		

EX Data

STUDYID	DOMAIN	USUBJID	EXSEQ	EXTRT	EXDOSE	EXDOSU	EXSTDTC	EXENDTC
CDISC01	EX	001	1	drugA	20	MG	2010-01-09	2010-01-09
CDISC01	EX	001	2	drugA	20	MG	2010-01-10	2010-01-13
CDISC01	EX	001	3	drugB	20	MG	2010-02-09	2010-02-09
CDISC01	EX	001	4	drugB	20	MG	2010-02-10	2010-02-13
CDISC01	EX	001	5	drugC	20	MG	2010-03-09	2010-03-09
CDISC01	EX	001	6	drugC	20	MG	2010-03-10	2010-03-13
CDISC01	EX	001	7	drugD	20	MG	2010-04-10	2010-04-10
CDISC01	EX	001	8	drugD	20	MG	2010-04-11	2010-04-14
CDISC01	EX	002	1	drugD	20	MG	2010-01-11	2010-01-11
CDISC01	EX	002	2	drugD	20	MG	2010-01-12	2010-01-15
CDISC01	EX	002	3	drugC	20	MG	2010-02-11	2010-02-11
CDISC01	EX	002	4	drugC	20	MG	2010-02-12	2010-02-15

Figure 7: Sample Output (Showing only the treatment variables)

USUBJID	TRT01P	TRT02P	TRT03P	TRT04P	TRT01A	TRT02A	TRT03A	TRT04A
001	drugA	drugB	drugC	drugD	drugA	drugB	drugC	drugD
002	drugD	drugC	drugA	drugB	drugD	drugC	drugA	drugB
003	drugB	drugA	drugC	drugD	drugB	drugA	drugC	drugD
004	drugA	drugB	drugD	drugC	drugA	drugB	drugD	drugC
005	drugB	drugC	drugA	drugD				

TRT01SDT	TRT02SDT	TRT03SDT	TRT04SDT	TRT01EDT	TRT02EDT	TRT03EDT	TRT04EDT
09JAN2010	09FEB2010	09MAR2010	10APR2010	13JAN2010	13FEB2010	13MAR2010	14APR2010
11JAN2010	11FEB2010	11MAR2010	11APR2010	15JAN2010	15FEB2010	15MAR2010	15APR2010
14JAN2010	14FEB2010	14MAR2010	14APR2010	19JAN2010	19FEB2010	19MAR2010	20APR2010
18JAN2010	18FEB2010	18MAR2010	18APR2010	22JAN2010	22FEB2010	22MAR2010	22APR2010
.	.	.	.	.	.	.	.

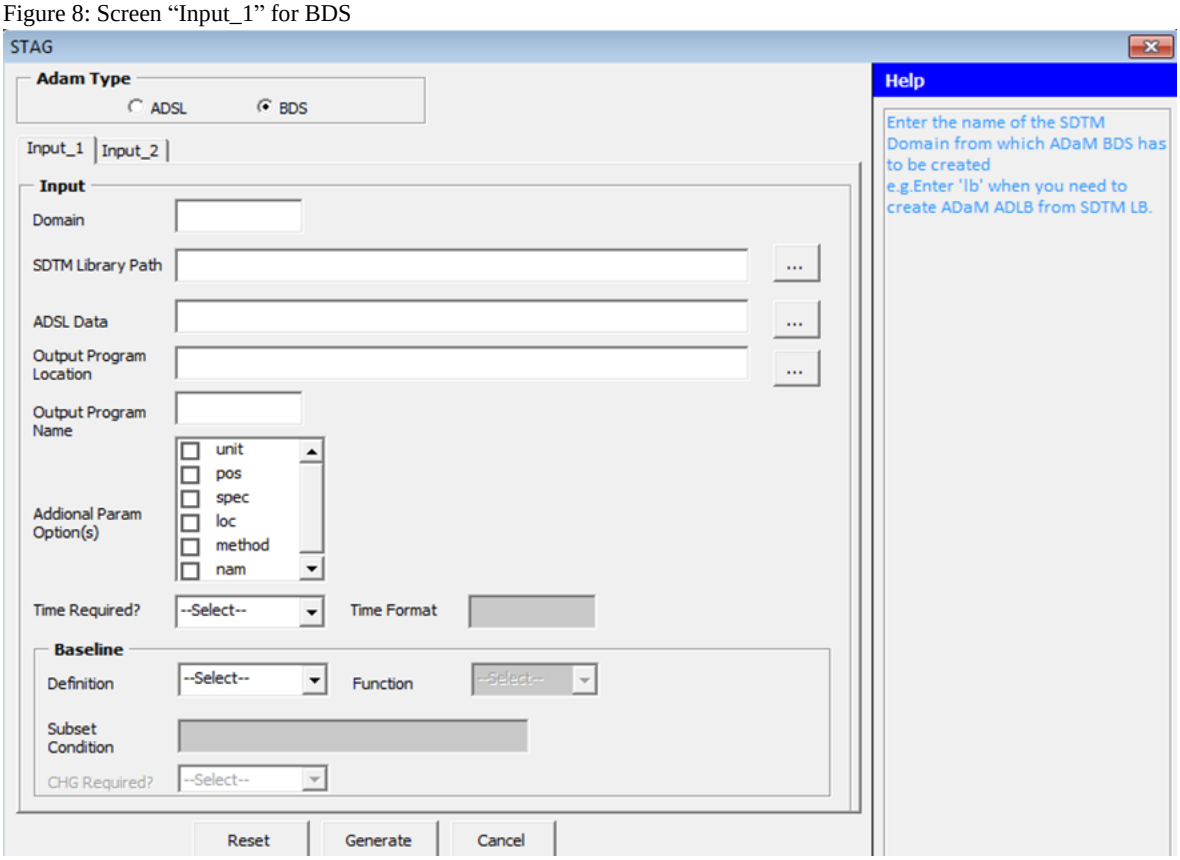
CREATING A STANDARD ADAM BDS DATASET

Once the user selects “BDS” radio button, a screen will appear and it will ask the user to enter some details. There are two screens “Input_1” and “Input_2” that will ask the user to enter certain inputs. Please note that currently in this tool, the capacity to generate a BDS is limited only to those ADaM datasets that can be created using Findings Class of SDTM Domains like LB, VS and so on. It cannot create more complex BDS that depend on multiple SDTM datasets like ADTTE (Time to Event) and so on. For example, ADLB and ADVS can be created using SDTM Datasets LB and VS respectively and most of the major basic ADaM BDS variables will be created.

The screen “Input_1” will ask the user to enter details as below:

1. Two letter domain abbreviation of the SDTM Domain required to create the BDS in question. For example, if you want to create a ADLB, please specify “LB” in this field.
2. The location of all SDTM Datasets
3. The location of ADSL dataset, since for creating any BDS an ADaM ADSL is expected to be ready.
4. The program file and location where it needs to be generated.
5. The options on how the user would like to define his Parameters i.e. variables PARAM(CD). For example, if unit is also required to along with a LBTESTCD to create a PARAM, please specify accordingly.
6. Whether for Time needs to be captured for analysis date and time related variables like ADT and ADTM.
7. Specify how baseline flag needs to be derived (Variable ABLFL)
8. Specify if you need to calculate variable related to Change from Baseline i.e. CHG.

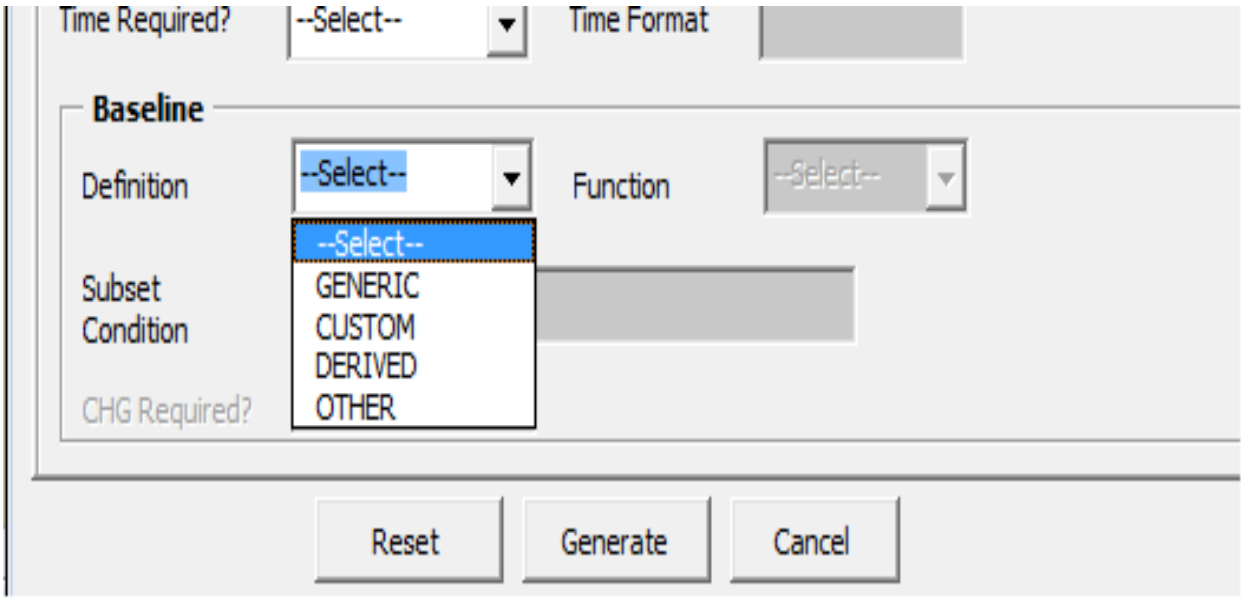
Figure 8: Screen “Input_1” for BDS



In “Input_1” screen, a close attention needs to be paid to the definition of Baseline. The tool allows three distinct options for creating ABLFL. (*Figure 9: Baseline*)

1. Select Generic, in case the definition of baseline is standard, which is last observed non-missing value before the first dose of protocol specified treatment.
2. Select “Custom”, in case the definition of baseline is not generic and rather it is based on some other sub setting condition. For example, Baseline is last observed non-missing value on or before enrolment, in which case specify a subset Condition like “<=ENRLDT” where ENRLDT is the enrolment date which should be present in the corresponding ADSL dataset.
3. Select “Derived”, in case you want to add a derived record for Baseline. For example, Baseline is a defined as average of all screening records, in which case specify Function as “AVERAGE” and Subset Condition as VISIT=1. In this case a variable “DTYPE” will also be created along with ABLDL (average of all AVAL where VISIT=1), because a derived record is being added.
4. Select “Other”, in case you would NOT like the tool to create a variable ABLFL.

Figure 9: Baseline

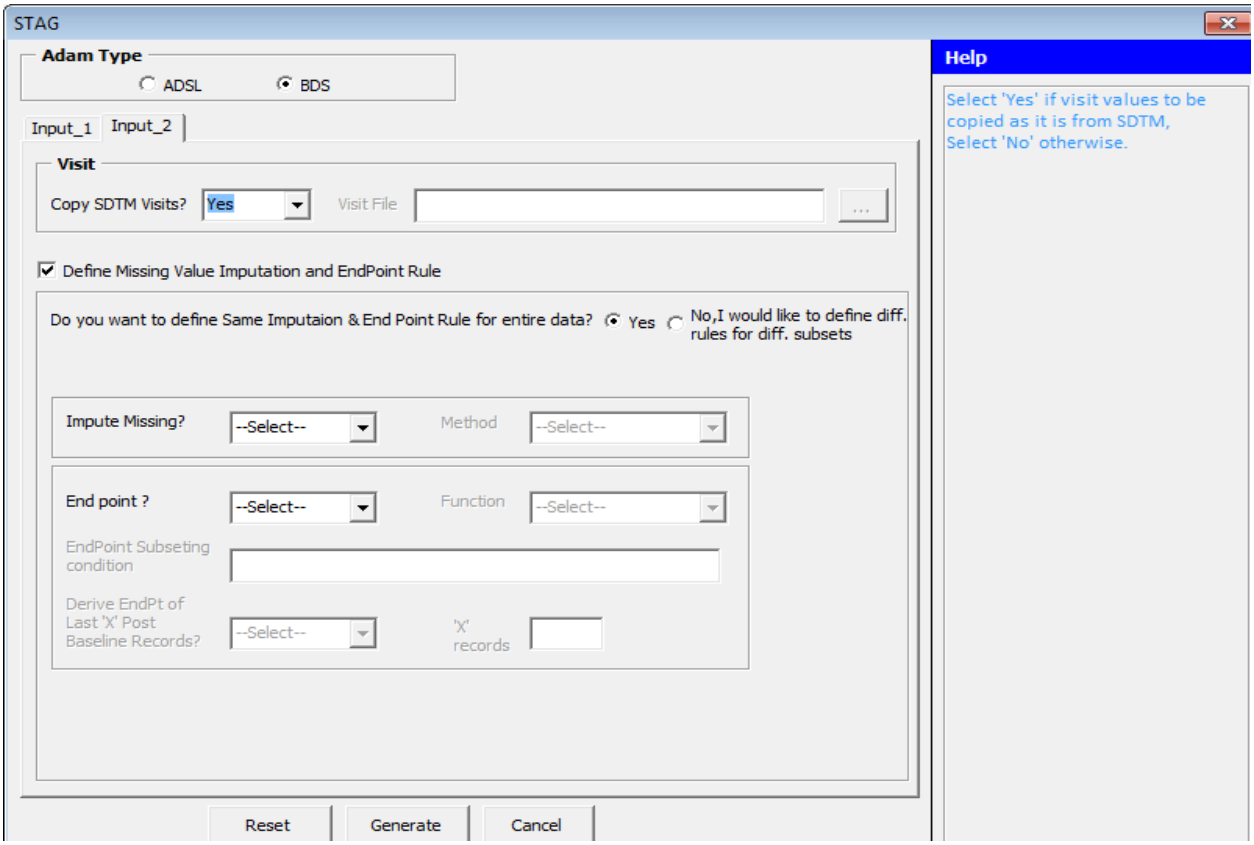


The screen “Input_2” will ask the user to enter details as below:

1. Specify how Visit Windowing and AVISIT and AVISITN needs to be calculated in the dataset. The user in this case needs to enter “Copy SDTM Visits?” as “No”, and also upload the Visit Windowing information and target date computation rules in an excel file with a standard format. In case, the User just needs to copy planned visits into AVISIT and AVISITN, the user should specify “Copy SDTM Visits?” as “Yes”, however it is very unlikely that Planned visits will be same as actual visits based on visit windowing.

2. Specify if the user wants to add derived records using DTYPE or add some Derived record for Endpoint. The current capacity of this tool allows for imputation of missing values by adding Derived records using Baseline Observation Carried Forward (DTYPE = BLOCF) and Last Observation Carried Forward (DTYPE = LOCF). The tools also has options to add Derived Records for Endpoints using 3 functions – Maximum, Minimum and Average.

Figure 10: Screen for "Input_2"



ADLB CREATION EXAMPLE

The sample input used in this example is a standard SDTM LB Dataset as shown in *Figure 11: Sample Input*. Please note the records for LBTESTCD = GLUC. There are three parameters that needs to be derived for LBTESTCD = GLUC:

1. Glucose Urine Qualitative with values like Negative, Trace and so on
2. Glucose Urine Quantitative measured in mmol/L
3. Glucose Blood Quantitative measured in mmol/L

Hence in this case we cannot simply assign PARAMCD as LBTESTCD as three distinct Parameters need to be derived for analysis. Notice how the user needs to specify these requirements in the UI Screen (*Figure 12: UI Screen)* and the resulting SAS® Code generated and the Output on running the SAS® Code in *Figure 13: Code Generated* and *Figure 14: Sample Output.*

Figure 11: Sample Input

STUDYID	USUBJID	LBSEQ	LBTESTCD	LBTEST	LBCAT	LBORRES	LBSTRESC	LBSTRESN	LBSTRESU	LBSPEC	VISIT
CDISC01	CDISC01.100008	1	BILI	Bilirubin	CHEMISTRY	0.4	6.8	6.8	umol/L	BLOOD	SCREEN
CDISC01	CDISC01.100008	2	BILI	Bilirubin	CHEMISTRY	0.3	5.1	5.1	umol/L	BLOOD	WEEK 24
CDISC01	CDISC01.100008	3	BUN	Blood Urea Nitro..	CHEMISTRY	26	9.28	9.28	mmol/L	BLOOD	SCREEN
CDISC01	CDISC01.100008	4	BUN	Blood Urea Nitro..	CHEMISTRY	18	6.43	6.43	mmol/L	BLOOD	WEEK 24
CDISC01	CDISC01.100008	5	GLUC	Glucose	CHEMISTRY	100	5.5	5.5	mmol/L	BLOOD	SCREEN
CDISC01	CDISC01.100008	6	GLUC	Glucose	CHEMISTRY	87	4.8	4.8	mmol/L	BLOOD	WEEK 24
CDISC01	CDISC01.100008	7	VITB12	Vitamin B12	CHEMISTRY	366	270	270	pmol/L	SERUM	SCREEN
CDISC01	CDISC01.100008	8	VITB9	Vitamin B9	CHEMISTRY	55.8	126.4	126.4	nmol/L	BLOOD	SCREEN
CDISC01	CDISC01.100008	9	HCT	Hematocrit	HEMATOLOGY	36.2	0.36	0.36		BLOOD	SCREEN
CDISC01	CDISC01.100008	10	HCT	Hematocrit	HEMATOLOGY	32.9	0.33	0.33		BLOOD	WEEK 24
CDISC01	CDISC01.100008	11	HGB	Hemoglobin	HEMATOLOGY	12.0	120	120	g/L	BLOOD	SCREEN
CDISC01	CDISC01.100008	12	HGB	Hemoglobin	HEMATOLOGY	11.3	113	113	g/L	BLOOD	WEEK 24
CDISC01	CDISC01.100008	13	LYM	Lymphocytes	HEMATOLOGY	1.68	1.68	1.68	10**9/L	BLOOD	SCREEN
CDISC01	CDISC01.100008	14	LYM	Lymphocytes	HEMATOLOGY	1.34	1.34	1.34	10**9/L	BLOOD	WEEK 24
CDISC01	CDISC01.100008	15	OCCBLD	Occult Blood	URINALYSIS	1+	1+	.		URINE	SCREEN
CDISC01	CDISC01.100008	16	OCCBLD	Occult Blood	URINALYSIS	NEGATIVE	NEGATIVE	.		URINE	WEEK 24
CDISC01	CDISC01.100008	17	PH	pH	URINALYSIS	8.0	8	8		URINE	SCREEN
CDISC01	CDISC01.100008	18	PH	pH	URINALYSIS	8.0	8	8		URINE	WEEK 24
CDISC01	CDISC01.100008	19	GLUC	Glucose	URINALYSIS	NEGATIVE	NEGATIVE	.		URINE	SCREEN
CDISC01	CDISC01.100008	20	GLUC	Glucose	URINALYSIS	TRACE	TRACE	.		URINE	WEEK 24
CDISC01	CDISC01.100008	21	GLUC	Glucose	URINALYSIS	10	0.6	0.6	mmol/L	URINE	SCREEN
CDISC01	CDISC01.100008	22	GLUC	Glucose	URINALYSIS	30	1.7	1.7	mmol/L	URINE	WEEK 24

Figure 12: UI Screen

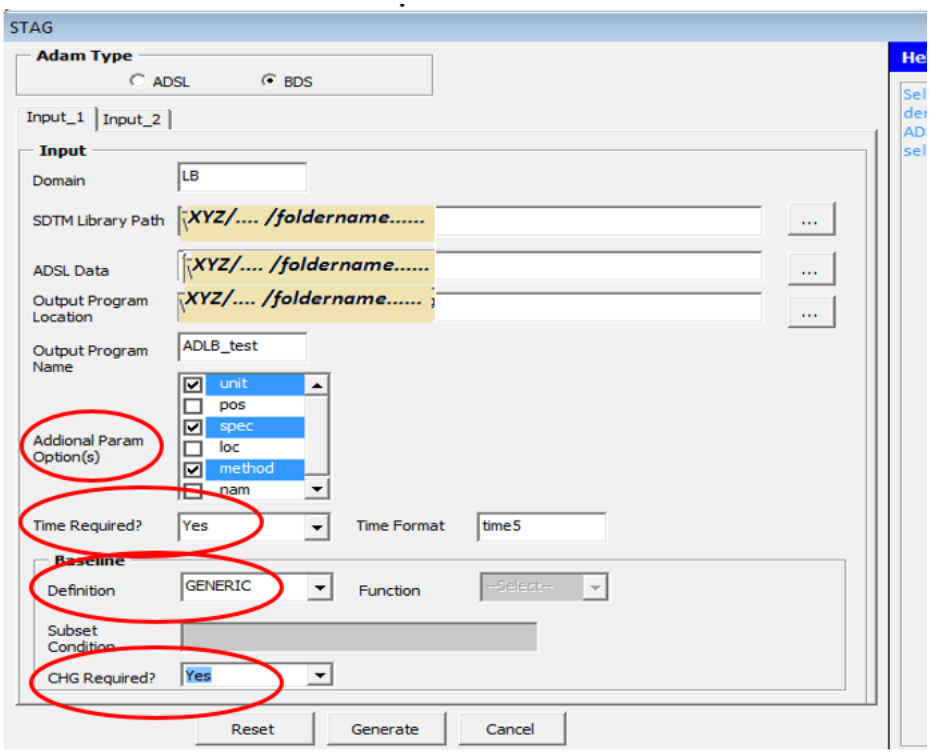


Figure 13: Code Generated

```
libname sdtm_dt 'XYZ/.... /foldername.....
libname adam_dt 'XYZ/.... /foldername.....
%include 'XYZ/.... /foldername.....
%include 'XYZ/.... /foldername.....
%include 'XYZ/.... /foldername.....
%include 'XYZ/.... /foldername.....

/*%generic_bds0x Macro Call*/
%generic_bds0x(sdtmlib=sdtm_dt, domain=LB, add= unit spec method,adsldata=adam_dt.adsl, basedef=GENE

/*Avisit Macro call*/
%avisit(indata=work.bds_LB, copy_sdtmvis=Yes);

data bds_out;
  set bds_vis_out;
run;

/*%generic_bds0y Macro Call*/
%generic_bds0y(inds=bds_out, sdtmlib=sdtm_dt, srcds =LB, impmissyn=no, endtptyn=no);

data pre_bds_LB;
  set bds_Out_der_recs;
run;
```

Figure 14: Sample Output

USUBJID	PARAM	PARAMN	PARAMCD	AVAL	AVALC	ADT	ATM	ADTM	ABLFL	BASE	BASEC	CHG
CDISC01.100008	Bilirubin (umol/L) (BLOOD)	1	BILI	6.8	6.8	15APR2003	11:20	15APR03:11:20:...	Y	6.8	6.8	0
CDISC01.100008	Bilirubin (umol/L) (BLOOD)	1	BILI	5.1	5.1	13OCT2003	11:55	13OCT03:11:55:...		6.8	6.8	-1.7
CDISC01.100008	Blood Urea Nitrogen (mmol/L) (BLOOD)	2	BUN	9.28	9.28	15APR2003	11:20	15APR03:11:20:...	Y	9.28	9.28	0
CDISC01.100008	Blood Urea Nitrogen (mmol/L) (BLOOD)	2	BUN	6.43	6.43	13OCT2003	11:55	13OCT03:11:55:...		9.28	9.28	-2.85
CDISC01.100008	Glucose (URINE) (DIPSTICK)	3	GLUC1	.	NEGATIVE	15APR2003	11:20	15APR03:11:20:...	Y	.	NEGATIVE	
CDISC01.100008	Glucose (URINE) (DIPSTICK)	3	GLUC1	.	TRACE	13OCT2003	11:55	13OCT03:11:55:...		.	NEGATIVE	
CDISC01.100008	Glucose (mmol/L) (BLOOD)	4	GLUC2	5.5	5.5	15APR2003	11:20	15APR03:11:20:...	Y	5.5	5.5	0
CDISC01.100008	Glucose (mmol/L) (BLOOD)	4	GLUC2	4.8	4.8	13OCT2003	11:55	13OCT03:11:55:...		5.5	5.5	-0.7
CDISC01.100008	Glucose (mmol/L) (URINE) (QUANT)	5	GLUC3	0.6	0.6	15APR2003	11:20	15APR03:11:20:...	Y	0.6	0.6	0
CDISC01.100008	Glucose (mmol/L) (URINE) (QUANT)	5	GLUC3	1.7	1.7	13OCT2003	11:55	13OCT03:11:55:...		0.6	0.6	1.1
CDISC01.100008	Hematocrit (BLOOD)	6	HCT	0.36	0.36	15APR2003	11:20	15APR03:11:20:...	Y	0.36	0.36	0
CDISC01.100008	Hematocrit (BLOOD)	6	HCT	0.33	0.33	13OCT2003	11:55	13OCT03:11:55:...		0.36	0.36	-0.03
CDISC01.100008	Hemoglobin (g/L) (BLOOD)	7	HGB	120	120	15APR2003	11:20	15APR03:11:20:...	Y	120	120	0
CDISC01.100008	Hemoglobin (g/L) (BLOOD)	7	HGB	113	113	13OCT2003	11:55	13OCT03:11:55:...		120	120	-7
CDISC01.100008	Lymphocytes (10**9/L) (BLOOD)	8	LYM	1.68	1.68	15APR2003	11:20	15APR03:11:20:...	Y	1.68	1.68	0
CDISC01.100008	Lymphocytes (10**9/L) (BLOOD)	8	LYM	1.34	1.34	13OCT2003	11:55	13OCT03:11:55:...		1.68	1.68	-0.34
CDISC01.100008	Occult Blood (URINE)	9	OCCBLD	.	1+	15APR2003	11:20	15APR03:11:20:...	Y	.	1+	.
CDISC01.100008	Occult Blood (URINE)	9	OCCBLD	.	NEGATIVE	13OCT2003	11:55	13OCT03:11:55:...		.	1+	.
CDISC01.100008	Vitamin B12 (pmol/L) (SERUM)	10	VITB12	270	270	15APR2003	11:20	15APR03:11:20:...	Y	270	270	0
CDISC01.100008	Vitamin B9 (nmol/L) (BLOOD)	11	VITB9	126.4	126.4	15APR2003	11:20	15APR03:11:20:...	Y	126.4	126.4	0
CDISC01.100008	pH (URINE)	12	PH	8	8	15APR2003	11:20	15APR03:11:20:...	Y	8	8	0
CDISC01.100008	pH (URINE)	12	PH	8	8	13OCT2003	11:55	13OCT03:11:55:...		8	8	0

CREATING DERIVED RECORDS

This tool in its current capacity can derived records in the following ways:

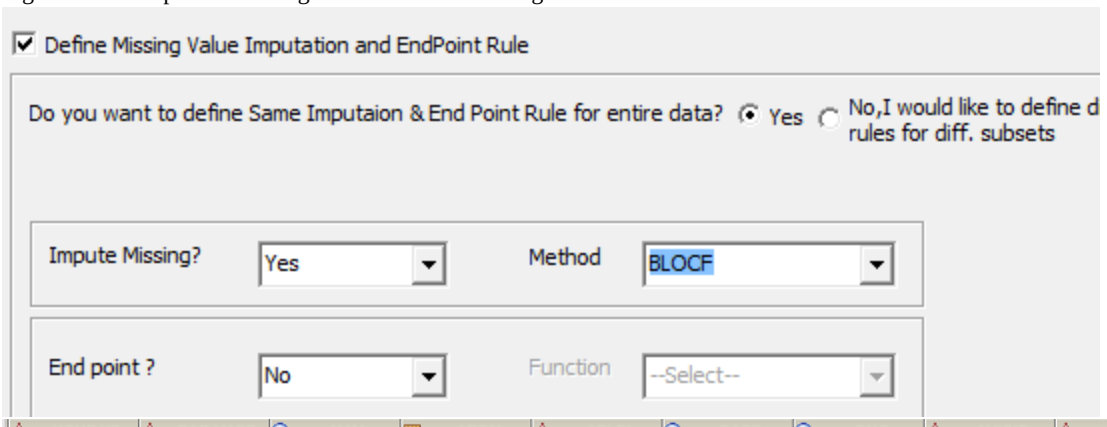
1. In case derived records need to be created for Imputing Missing values using Variable DTYPE. The tool allows two options – Last Observation Carried Forward (LOCF) and Baseline Observation Carried Forward (BLOCF). In this case the SDTM Variables from the parent records will be retained along other relevant variables like ADT, while the variables like

AVISIT are populated accordingly. The SAS® Codes working backend are robust and takes care of key aspects like Traceability.

2. In case Derived records need to be added for computing Endpoints using functions like Average, Maximum and Minimum.

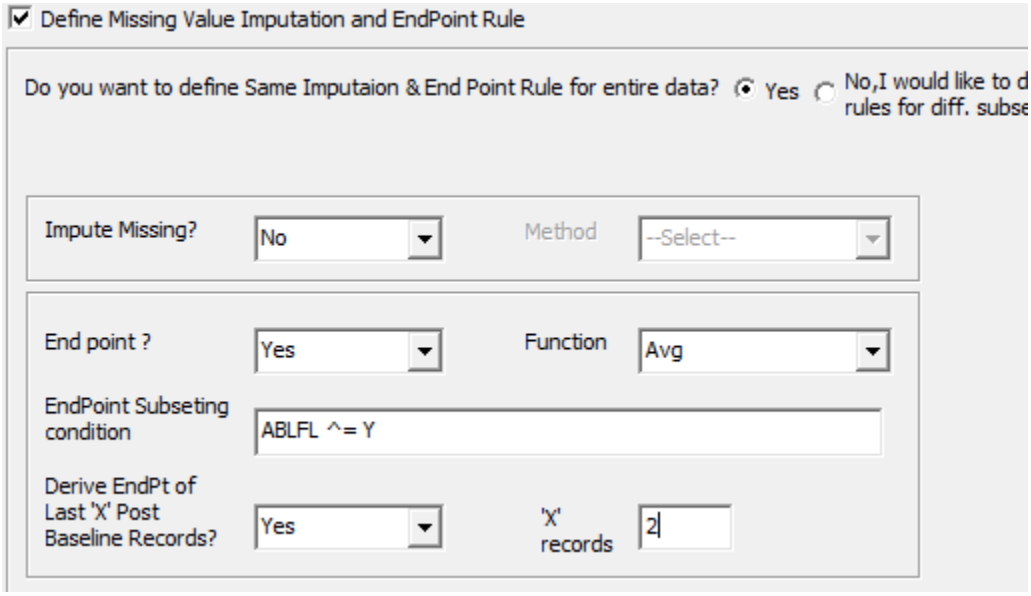
Examples below show the demonstration for usage.

Figure 15: Example for creating a Derived Record using BLOCF



USUBJID	PARAMCD	AVAL	ADTM	ABLFL	BASE	CHG	AVISIT	DTYPE
CDISC01.100008	HCT	0.36	15APR03:11:20:...	Y	0.36	0	SCREEN	
CDISC01.100008	HCT	0.33	13OCT03:11:55:...		0.36	-0.03	WEEK 24	
CDISC01.100008	HCT	0.33	13NOV03:11:55:...		0.36	-0.03	WEEK 28	
CDISC01.100008	HCT	0.45	13DEC03:11:55:...		0.36	0.09	WEEK 32	
CDISC01.100008	HGB	120	15APR03:11:20:...	Y	120	0	SCREEN	
CDISC01.100008	HGB	113	13OCT03:11:55:...		120	-7	WEEK 24	
CDISC01.100008	HGB	113	13NOV03:11:55:...		120	-7	WEEK 28	
CDISC01.100008	HGB	113	13DEC03:11:55:...		120	-7	WEEK 32	
CDISC01.200001	HCT	0.46	09SEP03:11:02:...	Y	0.46	0	SCREEN	
CDISC01.200001	HCT	0.48	02FEB04:08:58:...		0.46	0.02	WEEK 24	
CDISC01.200001	HCT	0.43	02MAR04:08:58:...		0.46	-0.03	WEEK 28	
CDISC01.200001	HCT	0.46	09SEP03:11:02:...		0.46	0	WEEK 32	BLOCF
CDISC01.200001	HGB	140	09SEP03:11:02:...	Y	140	0	SCREEN	
CDISC01.200001	HGB	135	02FEB04:08:58:...		140	-5	WEEK 24	
CDISC01.200001	HGB	135	02MAR04:08:58:...		140	-5	WEEK 28	
CDISC01.200001	HGB	135	02APR04:08:58:...		140	-5	WEEK 32	

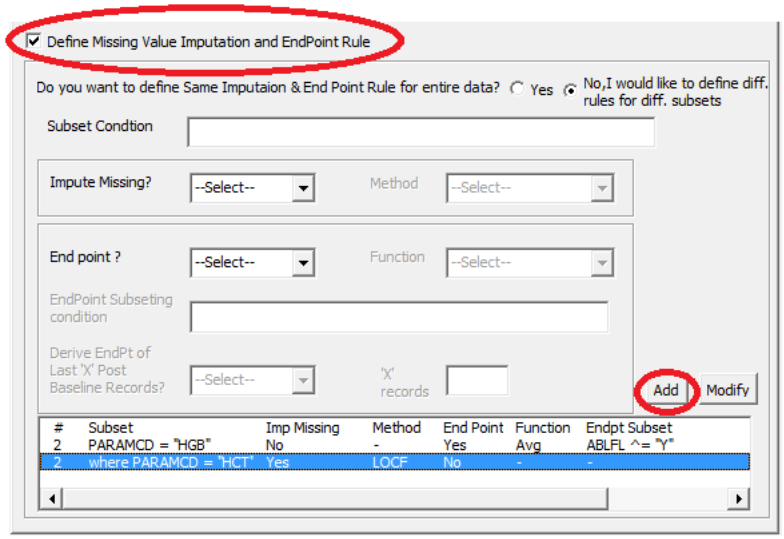
Figure 16: Example for creating a Derived Record for endpoint by taking the average of last 2 postbaseline records



USUBJID	PARAMCD	AVAL	ADTM	ABLFL	BASE	CHG	AVISIT	DTYPE
CDISC01.100008	HCT	0.36	15APR03:11:20:...	Y	0.36	0	SCREEN	
CDISC01.100008	HCT	0.33	13OCT03:11:55:...		0.36	-0.03	WEEK 24	
CDISC01.100008	HCT	0.33	13NOV03:11:55:...		0.36	-0.03	WEEK 28	
CDISC01.100008	HCT	0.45	13DEC03:11:55:...		0.36	0.09	WEEK 32	
CDISC01.100008	HCT	0.39	.		0.36	0.03	ENDPOINT	AVERAGE
CDISC01.100008	HGB	120	15APR03:11:20:...	Y	120	0	SCREEN	
CDISC01.100008	HGB	113	13OCT03:11:55:...		120	-7	WEEK 24	
CDISC01.100008	HGB	113	13NOV03:11:55:...		120	-7	WEEK 28	
CDISC01.100008	HGB	113	13DEC03:11:55:...		120	-7	WEEK 32	
CDISC01.100008	HGB	113	.		120	-7	ENDPOINT	AVERAGE
CDISC01.200001	HCT	0.46	09SEP03:11:02:...	Y	0.46	0	SCREEN	
CDISC01.200001	HCT	0.48	02FEB04:08:58:...		0.46	0.02	WEEK 24	
CDISC01.200001	HCT	0.43	02MAR04:08:58:...		0.46	-0.03	WEEK 28	
CDISC01.200001	HCT	0.455	.		0.46	-0.005	ENDPOINT	AVERAGE
CDISC01.200001	HGB	140	09SEP03:11:02:...	Y	140	0	SCREEN	
CDISC01.200001	HGB	135	02FEB04:08:58:...		140	-5	WEEK 24	
CDISC01.200001	HGB	135	02MAR04:08:58:...		140	-5	WEEK 28	
CDISC01.200001	HGB	135	02APR04:08:58:...		140	-5	WEEK 32	
CDISC01.200001	HGB	135	.		140	-5	ENDPOINT	AVERAGE

In certain scenarios, it may be needed to create derived records differently for different parameters depending on analysis requirements. For example, for a Given ADLB, one may need to derive endpoint derived record for Hemoglobin and Impute missing value using LOCF for Hematocrit. The below figure shows how this needs to be done using this tool.

Figure 17: For HGB we need to create derived record by taking average of all postbaseline records, and for HCT derived records need to be created for imputing missing values using LOCF.



USUBJID	PARAMCD	AVAL	ADTM	ABLFL	BASE	CHG	AVISIT	DTYPE
CDISC01.200001	HCT	0.46	09SEP03:11:02:...	Y	0.46	0	SCREEN	
CDISC01.200001	HCT	0.48	02FEB04:08:58:...		0.46	0.02	WEEK 24	
CDISC01.200001	HCT	0.43	02MAR04:08:58:...		0.46	-0.03	WEEK 28	
CDISC01.200001	HCT	0.43	02MAR04:08:58:...		0.46	-0.03	WEEK 32	LOCF
CDISC01.200001	HGB	140	09SEP03:11:02:...	Y	140	0	SCREEN	
CDISC01.200001	HGB	135	02FEB04:08:58:...		140	-5	WEEK 24	
CDISC01.200001	HGB	135	02MAR04:08:58:...		140	-5	WEEK 28	
CDISC01.200001	HGB	135	02APR04:08:58:...		140	-5	WEEK 32	
CDISC01.200001	HGB	135	.		140	-5	ENDPOINT	AVERAGE

CONCLUSION

The tool is currently undergoing internal testing and will then enter a pilot phase. It is particularly well suited for creating the ADSL (Subject-level analysis dataset) especially for common forms of study design with protocol specified exposure i.e. SDTM EX dataset should be present. The tool is also able to handle Basic Data Structure (BDS) ADaMs that are derived out of "General Observation Class Findings" SDTM Domains. However, currently it is not able to handle more complex BDS ADaMs (sourced from multiple SDTM Datasets e.g. ADTTE or complex efficacy) in a straightforward way. STAG could save time by minimizing the data-driven programming steps that are written to derive common ADaM variables used in almost in any analysis.

REFERENCES

ADaM Implementation Guide V1.1

CONTACT INFORMATION

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

Diganta Bose, Senior Team Lead – Statistical Programming Services,
Cytel Statistical Software and Services Pvt Ltd
Email: diganta.bose@cytel.com