PharmaSUG 2013 - Paper AD08

Just Press the Button Generation of SAS Code to Create Analysis Datasets directly from an SAP Can it be Done?

Endri Endri, Berlin, Germany
Rowland Hale, inVentiv Health Clinical, Berlin, Germany

ABSTRACT

The CDISC SDTM and ADaM models are rapidly becoming adopted as the data standard for clinical trials and submissions of clinical trial data to the FDA. ADaM datasets in particular form an integral part of clinical study analysis and require significant data derivation to fulfil the needs of table, figure and listing (TFL) requirements.

Since the move towards standardization is nothing new, are we now able to obtain not only SDTM and ADaM datasets but a full study analysis at the mere press of a button? The answer is of course still "No, not yet". The reason is that much still depends on the study design and the structure of the data, the derivation rules or maybe just a small footnote which needs to be printed on a particular table.

The paper goes on to explain how it may be possible to “translate” information written in a Statistical Analysis Plan (SAP) (such as imputation rules, population flag, baseline flag, etc.) to generate SAS code which will produce the analysis dataset. Using a text analysis algorithm a SAS macro will attempt to identify the requirements and determine the source data needed for creation of the analysis variables.

After generation of the SAS programs for the analysis, the statistical programmer or analyst just needs to review the resulting SAS code and make any required adjustments prior to validation, the aim being to help all of us as statistical programmers or analysts in our daily work and give us more time for validation and review!

INTRODUCTION

The primary aim of this paper is to discuss the possibility of automating the “translation” of information given in an SAP and in the TFL specifications directly into SAS programs for the statistical analysis.

The way to implement this automation is to carry out a “text analysis” of the (non-standard) SAP text and mock tables using a string comparison method to seek out the relevant information and utilise it in conjunction with the SDTM datasets to produce a final analysis.

A further challenge is that many pharmaceutical companies and CROs already have their own standard macros which they use to analyse their clinical data and which they will probably wish to continue to use. The code generation solution discussed in this paper should bear this in mind and make provision for the continued use of in-house macro libraries wherever possible.

So the fundamental question is whether it is possible to effect a comprehensive “detection” of data and requirements and automatically gain an understanding of the clinical data regardless of data structure to such an extent that a proper study analysis in full accordance with information provided in the SAP can be carried out.

SCOPE OF AUTOMATION PROCESS

An automated process such is this one will always have certain expectations and assumptions, and our system is no different in this respect. The main expectations are as follows:

- SDTM data structure

  SDTM is the data standard for clinical data. In theory it provides consistency across studies and projects, but in reality there are always small differences in the SDTM datasets within pharma companies or CROs and across therapeutic, project or study areas, such as additional variables in supplementary datasets and slight differences in formats, codelists and variable labels, etc., which must be handled by a process of this kind.

- ADaM datasets

  The purpose of ADaM is to provide “analysis-ready” datasets for “one proc away” analysis of clinical data. This means that an ADaM dataset should contain all variables and records for the analysis without any need for additional pre-processing to merge or derive data prior to producing a table or figure. This also means that ADaM datasets by their very nature need to retain flexibility, so it is not easy to achieve a consistent structure across studies.

  So the question here is how to automatically establish which ADaM variables need to be derived for the analysis and what the derivation rules (such as input variable, data handling rules, if/else conditions, etc.) for those variables are.

- Statistical Analysis Plan, TFL specifications and mock tables

  SAPs tend to become ever more consistent within a company. This includes not only the structure but in many cases also the text itself. However, different SAPs will never be exactly the same as they are always written using "free text" to one extent or another. And it is in the sections of the SAP which are most important for our automation exercise that free text is most prevalent: derivation rules, definition flags for baseline, population, analysis flag, rules for missing results, etc.

  Just like SAPs, TFL specifications and mock tables tend towards standardisation within pharma companies, and these are also largely written using free text.

  So the question is how to ”read” this non-standard, free text and “translate” it into meaningful instructions for an automated process. For example, how do we detect which variables are required for the TFLs, and how do we recognise what type of analysis is required (e.g. frequencies, summary statistics, graphs, etc.) for each requested TFL?

- SAS programs as output

  SAS code generated by the process must of course adhere to standards and SOPs, make appropriate use of SAS procedures and functions as well as available standard macros which carry out the necessary calculations and derivations and generate output such as frequencies, summaries, incidences, be well structured and clearly commented, contain the required header information, etc., and at the end be as close to validation-ready as possible.

Of course there are many other points for consideration when it comes to automating an entire statistical analysis. These include:

- Initialization of the system and environment
- Definition of global and local macro variables
- Definition of standard footnotes
- Definition of standard and study-specific formats for the study analysis, e.g. a special format for the layout of a figure
- Creation of temporary macros to avoid code repetition, e.g. demographics tables repeated for several populations

Whilst *all* of these points have to be borne in mind at the outset when developing such a system, this paper will concentrate on the major questions to be considered for an automated statistical analysis system.

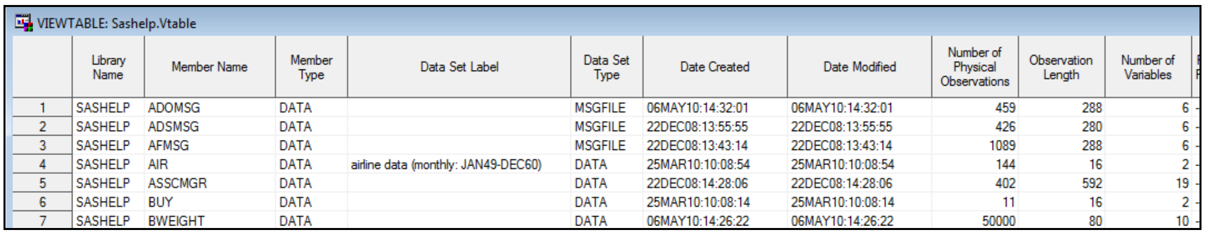
TIPS AND TRICKS FOR THE SOLUTION

ANALYSIS OF THE SOURCE DATA

The purpose of analysing the source data is to establish which SDTM datasets are present, and to establish what variables - attributes (label, length, type, numeric, format, etc.) and all – they contain. In some cases it should also be considered that some companies will have modified their SDTM datasets and added company or study specific features to the SDTM datasets.

All relevant information about the SDTM datasets can be found with the help of the SASHELP library and / or the CONTENTS procedure and can be saved as "metadata" for further automation processing.

Datasets present in a particular library can be found in the SASHELP.VTABLE data view. Here we can find such information as dataset names and numbers of observations.

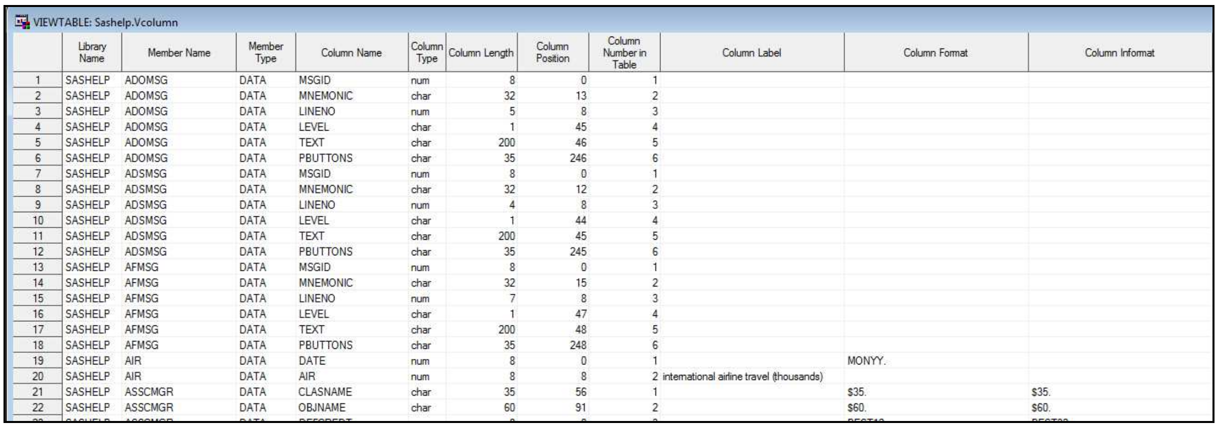


VIEWTABLE: Sashelp.Vtable

	Library Name	Member Name	Member Type	Data Set Label	Data Set Type	Date Created	Date Modified	Number of Physical Observations	Observation Length	Number of Variables
1	SASHELP	ADOMSG	DATA		MSGFILE	06MAY10:14:32:01	06MAY10:14:32:01	459	288	6
2	SASHELP	ADSMSG	DATA		MSGFILE	22DEC08:13:55:55	22DEC08:13:55:55	426	280	6
3	SASHELP	AFMSG	DATA		MSGFILE	22DEC08:13:43:14	22DEC08:13:43:14	1089	288	6
4	SASHELP	AIR	DATA	airline data (monthly: JAN49-DEC60)	DATA	25MAR10:10:08:54	25MAR10:10:08:54	144	16	2
5	SASHELP	ASSCMGR	DATA		DATA	22DEC08:14:28:06	22DEC08:14:28:06	402	592	19
6	SASHELP	BUY	DATA		DATA	25MAR10:10:08:14	25MAR10:10:08:14	11	16	2
7	SASHELP	BWEIGHT	DATA		DATA	06MAY10:14:26:22	06MAY10:14:26:22	50000	80	10

Figure 1. Sashelp.vtable dataset

SASHELP.VCOLUMN gives us information about the variables and their attributes in each dataset present in a particular library.

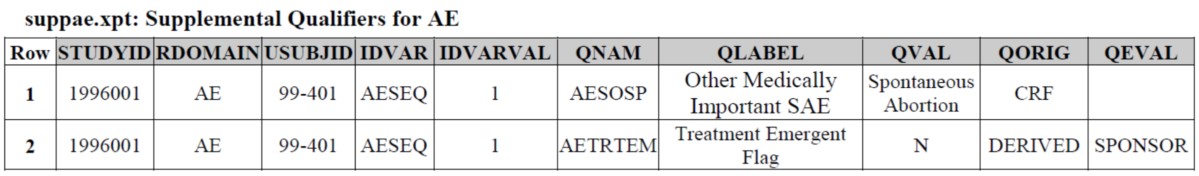


VIEWTABLE: Sashelp.Vcolumn

	Library Name	Member Name	Member Type	Column Name	Column Type	Column Length	Column Position	Column Number in Table	Column Label	Column Format	Column Informat
1	SASHELP	ADOMSG	DATA	MSGID	num	8	0	1			
2	SASHELP	ADOMSG	DATA	MNEMONIC	char	32	13	2			
3	SASHELP	ADOMSG	DATA	LINENO	num	5	8	3			
4	SASHELP	ADOMSG	DATA	LEVEL	char	1	45	4			
5	SASHELP	ADOMSG	DATA	TEXT	char	200	46	5			
6	SASHELP	ADOMSG	DATA	PBUTTONS	char	35	246	6			
7	SASHELP	ADSMSG	DATA	MSGID	num	8	0	1			
8	SASHELP	ADSMSG	DATA	MNEMONIC	char	32	12	2			
9	SASHELP	ADSMSG	DATA	LINENO	num	4	8	3			
10	SASHELP	ADSMSG	DATA	LEVEL	char	1	44	4			
11	SASHELP	ADSMSG	DATA	TEXT	char	200	45	5			
12	SASHELP	ADSMSG	DATA	PBUTTONS	char	35	245	6			
13	SASHELP	AFMSG	DATA	MSGID	num	8	0	1			
14	SASHELP	AFMSG	DATA	MNEMONIC	char	32	15	2			
15	SASHELP	AFMSG	DATA	LINENO	num	7	8	3			
16	SASHELP	AFMSG	DATA	LEVEL	char	1	47	4			
17	SASHELP	AFMSG	DATA	TEXT	char	200	48	5			
18	SASHELP	AFMSG	DATA	PBUTTONS	char	35	248	6			
19	SASHELP	AIR	DATA	DATE	num	8	0	1		MONYY.	
20	SASHELP	AIR	DATA	AIR	num	8	8	2	international airline travel (thousands)		
21	SASHELP	ASSCMGR	DATA	CLASNAME	char	35	56	1		$35.	$35.
22	SASHELP	ASSCMGR	DATA	OBJNAME	char	60	91	2		$60.	$60.

Figure 2. Sashelp.vcolumn dataset

Furthermore, existing "supplementary variables" in supplementary SDTM datasets are saved in the QNAM (Qualifier Variable Name) and QLABEL (Qualifier Variable Label) variables, for example:



suppae.xpt: Supplemental Qualifiers for AE

Row	STUDYID	RDOMAIN	USUBJID	IDVAR	IDVARVAL	QNAM	QLABEL	QVAL	QORIG	QEVAL
1	1996001	AE	99-401	AESEQ	1	AESOSP	Other Medically Important SAE	Spontaneous Abortion	CRF	
2	1996001	AE	99-401	AESEQ	1	AETRTEM	Treatment Emergent Flag	N	DERIVED	SPONSOR

Figure 3. Excerpt from SDTM IG 3.1.3 p.262

Gathering all of this information and saving it as metadata, assuming that such metadata does not exist already, is straightforward regardless of the structure of the SDTM database and invaluable when it comes to automating tasks such as the one discussed in this paper.

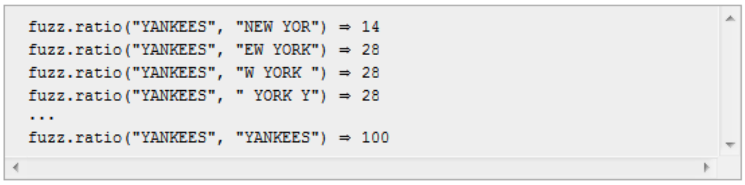
ANALYSIS OF (FREE) TEXT

The SAP contains information about the clinical study and is fundamental for the analysis. It contains derivation and data handling rules, and defines which tables, figures and listings are to be created. In some cases a SAP can also include mock tables, which define the layout of the requested tables.

SAPs contain much standardized text and generally have a standardized structure, but nowadays SAPs are still written using free text. Reading and analysing such a non-standard SAP programmatically presents something of a challenge for an automated process.

The solution for analysing free text is to use a string comparison method. This method is available in other programming languages such as Python where it is known as the “Fuzzy-Wuzzy method”. This method compares characters and returns the similarity of two pieces of text as a percentage. SAS® Text Miner also offers similar “string-comparison-method” functionality.

The macros for this paper were developed in Base SAS, which does not provide a built-in text comparison functionality, so a SAS macro with a string comparison method had to be developed to calculate the similarity of text strings and enabled the information relevant to our process to be captured. Furthermore, the fact that we developed the method ourselves gave us much more flexibility in terms of the comparison algorithm.



```
fuzz.ratio("YANKEES", "NEW YOR") ⇒ 14
fuzz.ratio("YANKEES", "EW YORK") ⇒ 28
fuzz.ratio("YANKEES", "W YORK ") ⇒ 28
fuzz.ratio("YANKEES", " YORK Y") ⇒ 28
...
fuzz.ratio("YANKEES", "YANKEES") ⇒ 100
```

Figure 4. Example of fuzzy wuzzy method in Python

INTERNAL DICTIONARY

Fundamental to the solution is the implementation of an internal dictionary. The dictionary has the following sections:

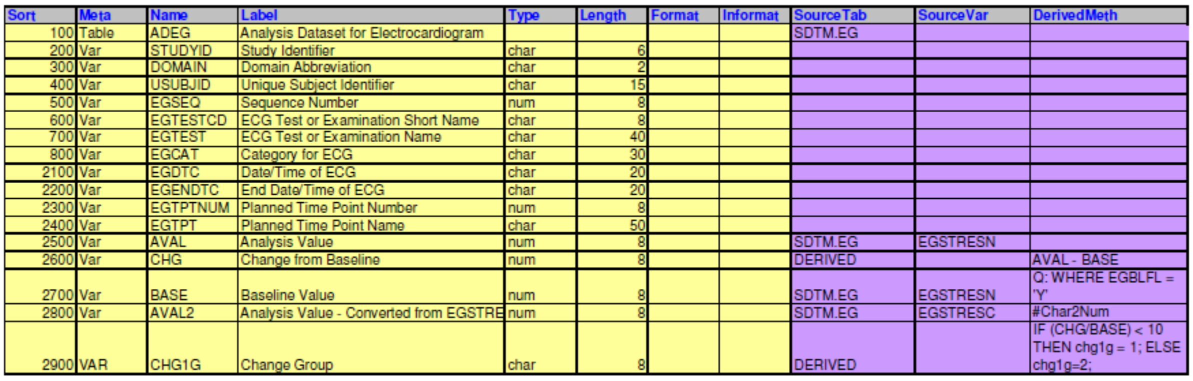
- System keywords
  Standard SAS functions and procedures are saved into this internal dictionary to facilitate their use in the generation of output SAS code.
- Definition of standard macros
  Standard macros are also saved in this internal dictionary. Using text replacement, the process “auto-replaces” the parameters of macro calls, and in turn enables standard macros to be used in the generation of the final SAS code.
- Definition of synonyms
  In several cases synonyms are needed to recognise “unknown” words in the SAP and mock tables, e.g “subject” has the same meaning as “patient”.
- Memory function
  Special definitions, such as the definition of baseline, are needed in every study. The memory function gives us the ability to save definitions encountered together with their solutions, so that they can be used in other studies with minimal additional pre-processing.

The internal dictionary solution provides a very effective way to save definitions and keywords. This approach mirrors artificial intelligence algorithms, which allow the whole system to get smarter and faster in terms of doing the analysis. This is only possible thanks to the memory function and recall method.

SAS CODE GENERATOR

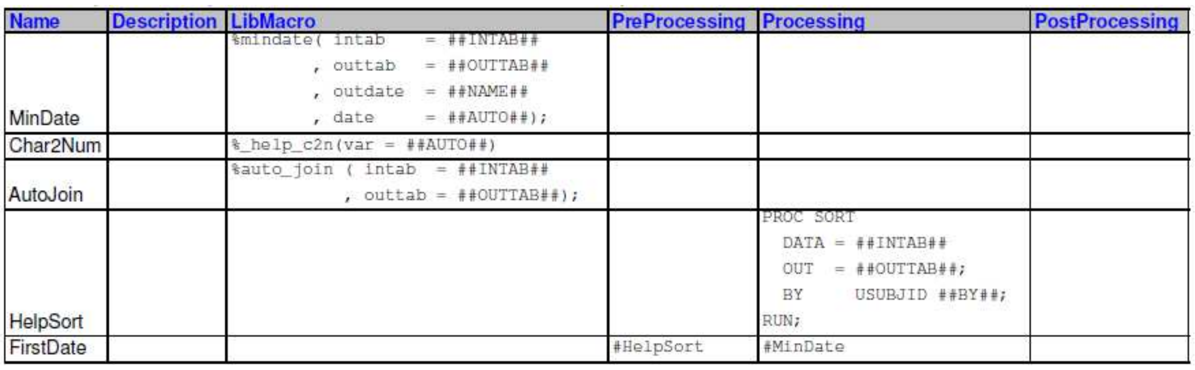
The code generation method was presented at the 2011 PhUSE Conference held in Brighton, UK (DH07 – Endri Endri; Rowland Hale: Much ADaM about Nothing - a PROC Away in a Day). Further information about the SAS program generator engine can be found in this paper, which describes in detail how ADaM data sets can be programmed using Microsoft Excel.

With auto-text functionality (##text##) a code generator engine is able to generate the pre-defined program header, use existing standard macros, utilise SAS functions and procedures, etc.



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;

Figure 5. Excerpt from ADEG metadata



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	

Figure 6. Excerpt of definition of standard macros

EXAMPLE OF AUTOMATION PROCESS

The following examples illustrate the automation process:

1. REQUESTED TABLE : "... (MALE ONLY)"

Say a particular table needs to contain only the male subjects in a study. As previously described, the automation process first attempts to "understand" the data in question by extracting the required information about the data structure, including the study specify format.

```
* Study specify format;
PROC FORMAT LIB = WORK;
  VALUE _sex
     1 = 'M'
     2 = 'F';
QUIT;

* Sample unknown data structure;
DATA random;
  ATTRIB subj    FORMAT = $20.
         gender  FORMAT = _sex.;
  subj = '1'; gender = 1; OUTPUT;
  subj = '2'; gender = 2; OUTPUT;
RUN;
```

Output 1. Example of dataset with undefined structure and study specify format

Using all the information gathered, including that from the requirement, the solution is reached as follows:

- The internal dictionary of pre-defined keywords is used. Here the word "only" is defined as a system keyword which translates to "IF .... EQ .... ".
- "Male" is not defined as a system keyword and is therefore analyzed using the "string-comparison-method".
- Resulting code: `IF gender EQ 1;`

2. POPULATION DEFINITION

Say the safety population is defined as *"... if he/she is randomized to a treatment group and has taken at least one unit of the study medication and has post-treatment safety data available."*, here the CDISC SDTM datasets are used for the requirement analysis.

Solution:

- "*randomized*" can be found in the DS dataset.
- "*treatment group*" information can be found in variable ARMCD in the DM dataset
- "*at least*" is defined as system keyword, which will be translated into : `HAVING MIN (##var##) >= ##`
- "*study medication*" can be found in the EX dataset
- "*post-treatment safety data*" information can be found in the VS dataset

3. SAS CODE GENERATOR

All the information captured from an "analysis" of the CDISC SDTM datasets and of the (free) text in the SAP is converted into SAS code.

The following is an example of a SAS program produced by the SAS code generator used (as described in our previous PhUSE paper):

```
/**********************************************************
* 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);
```

Output 2. Derivation of ADEG – automatically created using %adam_gen macro

CONCLUSION

The aim of this exercise was to achieve, as far as possible, a fully automated study analysis – from SAP, TFL specifications and mock tables to final analysis – by automatically capturing the required information from these documents and applying it to an SDTM clinical database through the use of complex algorithms.

The development of this system, which took into consideration all of the points discussed in this paper and used nothing other than Base SAS and SAS Macro, has now reached final testing. A test generation of SAS programs to create around 100 TFLs using the system took in the region of 8 hours processing time.

Although the automation process may "misunderstand" certain aspects of the SAP, this process helps significantly with the programming task not least because the ability to adjust the SAS programs produced is retained. This is much more efficient than writing SAS programs manually from scratch. And if an SAP and TFL specifications are drawn up with such an automated system in mind, then the performance and time efficiency gains can only be enhanced.

The system is about to be tested by Mr. P. Jähnig (PJ Statistics, Berlin – Germany) and ICRC Weyer GmbH, a full-service Contract Research Organization (CRO) in Berlin – Germany.

Furthermore, the fact that the system comprises a number of individual tasks means that this is not an "all or nothing" process and it can be used to automate one or more of the following tasks, as required:

- Simple assignment
  i.e. for allocation of a CDISC Controlled Terminology codelist based on local laboratory parameter names
- Mapping and data derivation
  Example: SDTM and ADaM mapping
- Query and data validation
- Table, Figure and Listing programming

So the answer to the question "Can a full study analysis be done at the press of a button?" is indeed "No, not yet", but we have shown that it is possible to move some way towards this goal.

CONTACT INFORMATION

Your comments and questions are valued and encouraged.

Contact the author at:

Name	: Endri	Rowland Hale
Enterprise	: - - -	inVentiv Health Clinical
Address	: Berlin	Berlin
City,StateZIP	: Germany	Germany
Work Phone	: +49 17 620 625 237	+49 30 345 069 10
E-mail	: endri0501@yahoo.de	rowland.hale@inventivhealth.com
Profile	: http://de.linkedin.com/in/endri0501	http://www.inventivhealth.com/