Paper ET13

Rule-Based Auto-SDTM Mapping Software in Python Which Automatically Generates Executable SAS and R Programs

Carrie Zhang, ClinChoice Inc., Horsham, PA, USA
Bill Zhang, ClinChoice Inc., Horsham, PA, USA

ABSTRACT

The creation of CDISC SDTM datasets is often tedious and time-consuming, requiring repeated manual programming across studies. Such manual process may lead to inconsistency and non-compliance. To address these challenges, we developed a rule-based SDTM auto-mapping software using Python that leverages a knowledge library of metadata and generic mapping rules between raw datasets and SDTM datasets. The software parses predefined, reusable rules, integrates controlled terminology, and automatically generates executable SAS or R programs to produce final SDTM datasets. In addition, it generates SDTM metadata to facilitate review and regulatory submission.

This automated solution significantly improves efficiency, strengthens compliance with CDISC standards, enhances traceability, and reduces both manual effort and human error. The tool supports multiple EDC data sources as well as third-party vendor (TPV) data. This paper highlights the system design and implementation strategies, offering a practical framework for sponsors and CROs seeking scalable, compliant SDTM automation solutions.

INTRODUCTION

SDTM is a key requirement for regulatory submissions to the FDA, PMDA, and other global authorities. Despite being a well-defined standard, SDTM conversion remains one of the most time-consuming activities in clinical data management. Traditional SDTM creation workflows rely heavily on manual programming, introducing inefficiency, variability, and higher risk of errors. This paper presents a rule-driven SDTM automation software tool that generates executable SAS or R code and submission-ready metadata with Python implementation.

BACKGROUND AND MOTIVATION

Repetitive Patterns in SDTM Mapping: Domains such as AE, VS, and EX follow consistent structures across studies, creating opportunities for reuse.

Complexity of Evolving Standards: Frequent updates to SDTM IGs and NCI Controlled Terminology require continuous maintenance and governance.

Increasing Volume and Variety of Data: Clinical trials usually incorporate both EDC and third-party datasets (labs, ECG, PK), increasing mapping complexity.

Industry Intend: The industry is exploring different SDTM auto-mapping techniques, including SAS macros and AI assistants. We use rule-based automation with AI to achieve standardized, accurate, and traceable SDTM mapping. This rule-based approach provides a more general framework for generating SDTM domains automatically.

SOFTWARE DESIGN

TECHNOLOGY STACK

The software is built in **Python**, using packages like pandas, Lex-Yacc (sly), Excel tools, and AI language libraries.
Lex-Yacc (sly) serves as the mapping rule parser and compiler.
The Python scripts run from the command line for easy integration with web servers, AI Agents, and standalone applications.
All documents are provided in Excel format, following industry standards.
The software supports SAS and R integration. It can execute SAS or R scripts in the process flow

DOCUMENTS & LIRARY

The following documents and libraries are adopted and unified to support standardized SDTM mapping process.

a. CDSIC SDTM IGs & Controlled Terminology

CDISC SDTM IGs and Controlled Terminology documents are downloaded to support multiple versions of SDTM. These libraries are referred while generating study SDTM metadata file.

b. Sponsor SDTM Rulebook

SDTM rulebooks serve as reference libraries for configuring SDTM mapping in future studies, often having multiple versions to align with different SDTM IG versions.
Each rulebook mainly outlines the mapping rules from RAW variables to SDTM variables, typically using calculation expressions as described below:

RAW			MAPPING	SDTM			
Dataset	Variable	Label	Map Rule	Dataset	Variable	Label	Core
AE	STUDY	Study Code	COPY()	AE	STUDYID	Study Identifier	Req
AE			"AE"	AE	DOMAIN	Domain Abbreviation	Req
AE	STUDY SUBJECT	Study Code Enrolment Code	USUBJID()	AE	USUBJID	Unique Subject Identifier	Req
AE	AENO	AE Number	CONCAT('AE-', AENO)	AE	AESPID	Sponsor-Defined Identifier	Perm
AE	AETERM	Adverse Event Reported Term	COPY()	AE	AETERM	Reported Term for the Adverse Event	Req
AE	LLT_NAME	MedDRA Lowest Level Term Name	COPY()	AE	AELLT	Lowest Level Term	Exp
AE	LLT_CODE	MedDRA Lowest Level Term Code	COPY()	AE	AELLTCD	Lowest Level Term Code	Exp
AE	PT_NAME	MedDRA Preferred Term Name	COPY()	AE	AEDECOD	Dictionary-Derived Term	Req
AE	PT_CODE	MedDRA Preferred Term Code	COPY()	AE	AEPTCD	Preferred Term Code	Exp
AE	HLT_NAME	MedDRA High Level Term Name	COPY()	AE	AEHLT	High Level Term	Exp
AE	HLT_CODE	MedDRA High Level Term Code	COPY()	AE	AEHLTCD	High Level Term Code	Exp
AE	HLGTNAME	MedDRA High Level Group Term Name	COPY()	AE	AEHLGT	High Level Group Term	Exp
AE	HLGTCODE	MedDRA High Level Group Term Code	COPY()	AE	AEHLGTCD	High Level Group Term Code	Exp
AE	SOC_NAME	MedDRA System Organ Class Name	COPY()	AE	AEBODSYS	Body System or Organ Class	Exp
AE	SOC_CODE	MedDRA System Organ Class Code	COPY()	AE	AEBDSYCD	Body System or Organ Class Code	Exp
AE	SOC_NAME	MedDRA System Organ Class Name	COPY()	AE	AESOC	Primary System Organ Class	Exp
AE	SOC_CODE	MedDRA System Organ Class Code	COPY()	AE	AESOCCD	Primary System Organ Class Code	Exp
AE	AESEV AEC01SEV AEC02SEV	AE Intensity AE Intensity AE Intensity	MAX(UPDECODE(AESEV), UPDECODE(AEC01SEV), UPDECODE(AEC02SEV))	AE	AESEV	Severity/Intensity	Perm
AE	AESEVMAX	Maximum AE Intensity	UPDECODE()	AE	AESEV	Severity/Intensity	Perm
AE	AESER	Serious Adverse Event	UPDECODE()	AE	AESER	Serious Event	Exp
AE	AEACN AEC01AC	Action Taken, Investigational Product Action Taken, Investigational Product	if AEC01AC ne '' then 'MULTIPLE' else if AEACN = '4' then 'DRUG WITHDRAWN' else UPDECODE(AEACN)	AE	AEACN	Action Taken with Study Treatment	Exp

c. Raw Metadata Specification File

Raw metadata specification file could be Rave EDC ALS, Veeva raw metadata file, and other metadata in different formats.

d. Unified Study Raw Metadata (SRM)

Raw metadata specification files are combined with customized utilities to produce unified Study Raw Metadata (SRM) file.
SRM, in excel format, contains ‘RawDomains’, ‘RawVariables’ and ‘RawFormats’ spreadsheets.

Dataset	Module	Variable	Variable Label	Type	Collection Length	Format	Current Version	Source
AE	AE	STUDY	Study Code	C	11		AZ2403 2024-03-25	CRF
AE	AE	SUBJECT	Enrolment Code	C	8		AZ2403 2024-03-25	CRF
AE	AE	PART	Study Part Code	C	1		AZ2403 2024-03-25	CRF
AE	AE	LINE	Line Number	N	3		AZ2403 2024-03-25	CRF
AE	AE	AEYN	Any Adverse Event	C	6	$NY	AZ2403 2024-03-25	CRF
AE	AE	AENO	AE Number	N	3		AZ2403 2024-03-25	CRF
AE	AE	AETERM	Adverse Event Reported Term	C	200		AZ2403 2024-03-25	CRF
AE	AE	AESTDAT	Adverse Event Start Date	C	10		AZ2403 2024-03-25	CRF
AE	AE	AEENDAT	Adverse Event End Date	C	10		AZ2403 2024-03-25	CRF
AE	AE	AEOUT	Outcome of Adverse Event	C	6	$OUT	AZ2403 2024-03-25	CRF
AE	AE	AESER	Serious Adverse Event	C	6	$NY	AZ2403 2024-03-25	CRF
AE	AE	IP	Investigational Product	C	20		AZ2403 2024-03-25	CRF
AE	AE	AEACN	Action Taken, Investigational Product	C	2	$AEACN	AZ2403 2024-03-25	CRF
AE	AE	AESEVMAX	Maximum AE Intensity	C	6	$AESEV	AZ2403 2024-03-25	CRF
AE	AE	AEREL	Reasonable Possibility AE Caused by IP	C	6	$NY	AZ2403 2024-03-25	CRF
AE	AE	AEDIS	AE Caused Subject to Withdraw from Study	C	6	$NY	AZ2403 2024-03-25	CRF
AE	AE	AEPROSSP	Provoked by Specific Study Procedure	C	6	$NY	AZ2403 2024-03-25	CRF
AE	AE	AECONTRT	Concomitant or Additional Trtmnt Given	C	6	$NY	AZ2403 2024-03-25	CRF
AE	AE	LLT_CODE	MedDRA Lowest Level Term Code	N	8		AZ2403 2024-03-25	CRF

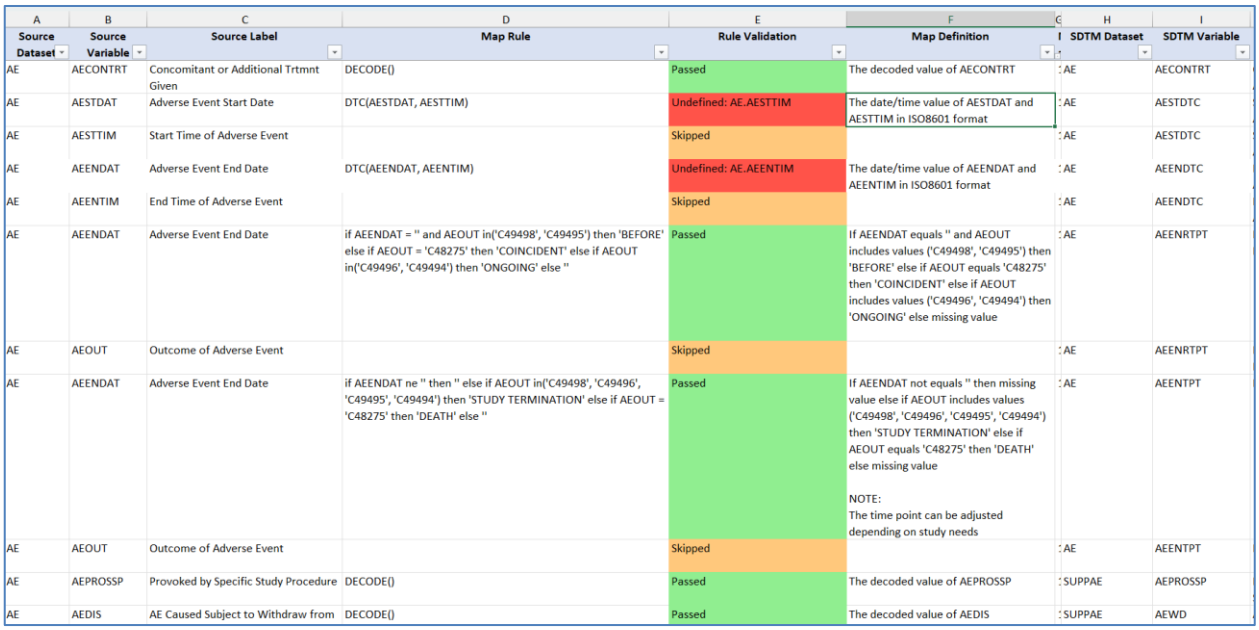
e1. Unified Study SDTM Metadata (SSM)

The software can read SRM and sponsor level SDTM metadata library to produce a unified Study SDTM Metadata (SSM) file. The format of SSM is compatible with the input for define.xml generation tool.
SSM, in excel format, contains 'Domains' and 'Variables' spreadsheets:

Dataset	Variable	Label	Type	Length	Core	Role	Format	Key	Seq	Origin
AE	STUDYID	Study Identifier	C	21	req	Identifier		1	1	Protocol
AE	DOMAIN	Domain Abbreviation	C	8	req	Identifier	(DOMAIN)		2	Derived
AE	USUBJID	Unique Subject Identifier	C	30	req	Identifier		2	3	Derived
AE	AESEQ	Sequence Number	N	8	req	Identifier			4	Derived
AE	AEGRPID	Group ID	C	40	perm	Identifier			5	Derived
AE	AEREFID	Reference ID	C	40	perm	Identifier			6	Derived
AE	AESPID	Sponsor-Defined Identifier	C	40	perm	Identifier			7	CRF
AE	AELNKID	Link ID	C	40	perm	Identifier			8	CRF
AE	AELNKGRP	Link Group ID	C	40	perm	Identifier			9	CRF
AE	AETERM	Reported Term for the Adverse Event	C	200	req	Topic			10	CRF
AE	AEMODIFY	Modified Reported Term	C	200	perm	Synonym Qualifier			11	Assigned
AE	AELLT	Lowest Level Term	C	200	exp	Variable Qualifier	MedDRA		12	Assigned
AE	AELLTCD	Lowest Level Term Code	N	8	exp	Variable Qualifier	MedDRA		13	Assigned
AE	AEDECOD	Dictionary-Derived Term	C	200	req	Synonym Qualifier	MedDRA	4	14	Assigned

e2. Unified Study Raw-to-SDTM Mapping Specification (SRSMS)

Study Raw-to-SDTM Mapping Specification (SRSMS) file is for RAW variable to SDTM variable mapping purpose.



Source Dataset	Source Variable	Source Label	Map Rule	Rule Validation	Map Definition	SDTM Dataset	SDTM Variable
AE	AECONTRT	Concomitant or Additional Trtmnt Given	DECODE()	Passed	The decoded value of AECONTRT	AE	AECONTRT
AE	AESTDAT	Adverse Event Start Date	DTC(AESTDAT, AESTTIM)	Undefined: AE.AESTTIM	The date/time value of AESTDAT and AESTTIM in ISO8601 format	AE	AESTDTC
AE	AESTTIM	Start Time of Adverse Event		Skipped		AE	AESTDTC
AE	AEENDAT	Adverse Event End Date	DTC(AEENDAT, AEENTIM)	Undefined: AE.AEENTIM	The date/time value of AEENDAT and AEENTIM in ISO8601 format	AE	AEENDTC
AE	AEENTIM	End Time of Adverse Event		Skipped		AE	AEENDTC
AE	AEENDAT	Adverse Event End Date	if AEENDAT = '' and AEOUT in('C49498', 'C49495') then 'BEFORE' else if AEOUT = 'C48275' then 'COINCIDENT' else if AEOUT in('C49496', 'C49494') then 'ONGOING' else ''	Passed	If AEENDAT equals '' and AEOUT includes values ('C49498', 'C49495') then 'BEFORE' else if AEOUT equals 'C48275' then 'COINCIDENT' else if AEOUT includes values ('C49496', 'C49494') then 'ONGOING' else missing value	AE	AEENRTPT
AE	AEOUT	Outcome of Adverse Event		Skipped		AE	AEENRTPT
AE	AEENDAT	Adverse Event End Date	if AEENDAT ne '' then '' else if AEOUT in('C49498', 'C49496', 'C49495', 'C49494') then 'STUDY TERMINATION' else if AEOUT = 'C48275' then 'DEATH' else ''	Passed	If AEENDAT not equals '' then missing value else if AEOUT includes values ('C49498', 'C49496', 'C49495', 'C49494') then 'STUDY TERMINATION' else if AEOUT equals 'C48275' then 'DEATH' else missing value NOTE: The time point can be adjusted depending on study needs	AE	AEENTPT
AE	AEOUT	Outcome of Adverse Event		Skipped		AE	AEENTPT
AE	AEPROSSP	Provoked by Specific Study Procedure	DECODE()	Passed	The decoded value of AEPROSSP	SUPPAE	AEPROSSP
AE	AEDIS	AE Caused Subject to Withdraw from	DECODE()	Passed	The decoded value of AEDIS	SUPPAE	AEWD

f. Predefined Mapping Rules

In the Mapping Rule file, the software has built-in support to utilize the following functions, SAS macros, expressions. These rules can be parsed and converted into SAS/R code.

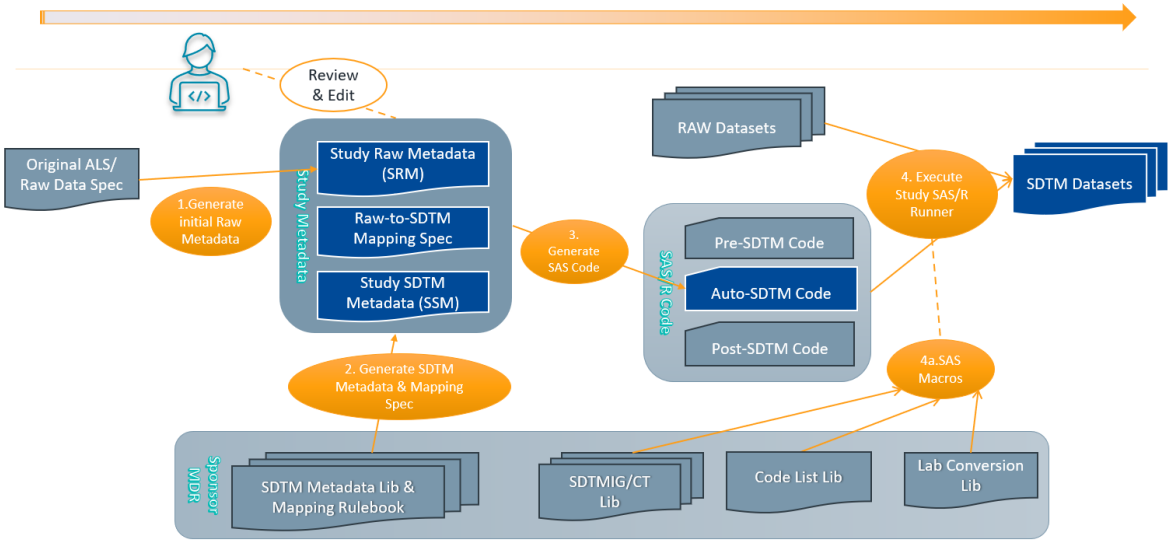
Mapping Expression	Alias	Description	Sample Expression	SAS Code
Expression Operators				
=, !=, >,<,>=,<=		Boolean operators for condition	Var2 = 'N' RAW1.VAR1>RAW2.VAR3	
+, -, *, /		Regular operators	RAW1.VAR1 * 2 *100 /3.2	
&, |, !, and, or, not		Logical operators	VAR1>VAR3 & VAR1<=VAR4 VAR1>VAR3 and VAR1<=VAR4	VAR1>VAR3 and VAR1<=VAR4
IN, NOT IN		in statement	var1 in ('1','A') var1 Not in ('2','B')	var1 in ('1','A') var1 Not in ('2','B')
Functions				
ISNULL	missing	Is missing	ISNULL(RAW.VAR1) Missing(RAW.VAR1)	Missing(RAW.VAR1)
IFC		If-else function	IF(ISNULL(RAW.VAR1), raw1.var1, raw2.var2) IF(RAW.VAR1 != "", raw1.var1)	
COPY() **COPY(<RAW variable>)**		Copy source value to target based on variable type	copy() copy(RAW.VAR4)	Char to Char: strip(<RAW Variable>) Num to Num:<RAW Variable>) Char to Num: input(<RAW Variable>, best.) Num to Char: strip(put(<RAW Variable>, best.))
USUBJID()		Concatenate STUDY, '/' and SUBJECT to USUBJID	USUBJID()	cats(STUDY, '/', SUBJECT)
DECODE() **DECODE(<RAW Variable>)** **DECODE(<Format>)** **DECODE(<RAW Variable>, <Format>)**	PUT,INPUT	Decode the RAW variable, or using RAW variable format, or using a specified format	DECODE() DECODE(<RAW Variable>) DECODE(<RAW Variable>, <Format>)	strip(putc(<RAW Variable>, raw_format(<RAW Variable>))) Decode the RAW variable using RAW variable format strip(put(<RAW Variable>, <Format>))

UPDECODE() **UPDECODE(<RAW Variable>)** **UPDECODE(<Format>)** **UPDECODE(<RAW Variable>, <Format>)**		Decode (with uppercase) the RAW variable, or using RAW variable format, or using a specified format	DECODE() DECODE(raw_var1) DECODE($NY) DECODE(raw_var2, $Y)	strip(putc(<RAW Variable>, raw_format(<RAW Variable>))) Decode the RAW variable using RAW variable format strip(put(<RAW Variable>, <Format>))
STRIP(), STRIP(Var1)		Convert to String	STRIP(PUT(RAW.VAR," BEST."))	STRIP(PUT(VAR)
VARLABEL(<RAW Variable>)		Set to the RAW variable label		strip(vlabel(<RAW Variable>))
UPCASE(<RAW Variable>)	TOUPPER	Set to the RAW variable value in upper case	UPCASE(RAW.CMTRT)	upcase(strip(<RAW Variable>))
SCAN		separate parts of a character value. (split)	**SCAN**(RAW.VAR1, 2, "T")	**SCAN**(RAW.VAR1, 2, "T")
FIND		To locate a substring within a string.	FIND(RAW.VAR1, "XXXX")	FIND(RAW.VAR1, "XXXX")
DELETE_IF(condition)		skip record if the condition is met	delete_if(AETERM=' ')	if AETERM='' then delete;
CATX(<Variable1>, '<String1>', <Variable2>, …)	Paste	Concatenates multiple character variables and/or strings	CATX(separator, RAW.OTHER,RAW.OTHERSP)	strip(<Variable1>)||<String1>||strip(<Variable2>)||…
CATS('<String1>', <Variable2>, …)	Paste0, cancat	Concatenates multiple character variables and/or strings	CATS(RAW.OTHER,RAW.OTHERSP)	strip(<Variable1>)||<String1>||strip(<Variable2>)||…
MIN(<Variable1/Value1>, <Variable2/Value2>, …)		Get the lowest value of the listed values		min(<Variable1/Value1>, <Variable2/Value2>, …)
Macros (%MACRO_NAME(...))				
%SEQ(<Variable1>, <Variable2>, …)		Creates the Sequence Number		%seq(dataset_layout=(<Variable1> <Variable2> …, outvar_seq=WORK.<SDTM Dataset>.<SDTM Variable>);
DTC() **DTC(<Date Variable)**	ISO_DATE	Creates the DTC from date variable		%dtc(INVAR_DAT=<Date Variable>, OUTVAR_DTC=<SDTM Variable>)
DTC(<Date Variable>, <Time Variable>)	ISO_DATETIME	Creates the DTC from date and time variables		%dtc(INVAR_DAT=<Date Variable>, INVAR_TIM=<Time Variable>, OUTVAR_DTC=<SDTM Variable>)

SOFTWARE IMPLEMENTATION

SDTM MAPING HIGH LEVEL FLOW

The SDTM mapping process flow includes multiple steps. Each step can be performed by software modules automatically. The programmer can review the output from each step and update the corresponding input files.



MAPPING STEPS

The SDTM mapping process flow contains the following modules:

1. Study Raw Metadata (SRM) Generator

Description: Generating study level SRM file from Raw Metadata Specification file
Input: Multiple Raw Metadata Specification files, Sponsor level metadata library, etc.
Output: SRM file with audit trail

2a. Study SDTM Metadata (SSM) Generator

Description: Generating study level SSM file from SRM, SDTM IGs, Sponsor level SDTM metadata library
Input: SRM, SDTM IGs, Sponsor level SDTM metadata library, older version of SSM.
Output: SSM file with audit trail

2b. SDTM Mapping Rule Generator

Description: Create raw to SDTM mapping rules by first matching existing variables; if not found, use an AI language model to identify the closest match based on label description.
Input: SRM, Sponsor level SDTM metadata library.
Output: Study Raw-to-SDTM Mapping Specification (SRSMS)

3a. Mapping Rule Parser

Description: Read and parse input mapping rules in order from SRSMS file
Input: Study Raw-to-SDTM Mapping Specification (SRSMS)
Output: Study rule mapping objects in Python, rule validation result

3b. SDTM Source Code Generator

Description: Generate SAS or R code from rule mapping objects in Python
Input: Study rule mapping objects in Python
Output: SAS or R source code

4. SDTM Dataset Mapping Runner

Description: Execute SAS/R code in Batch mode
Input: SAS/R code + Raw datasets, SAS macros, data formats
Output: SDTM datasets

SPONSOR CUSTIMIZATION

- Each SDTM domain can include pre- and post-process code for special handling, which may involve dataset-level SAS macros.
- The supported mapping rule definition can be customized by updating mapping rule parser
- Generated SAS/R code can be changed with updated SDTM Source Code Generator
- The variable formats are specified in the SDTM metadata spreadsheet that can be easily modified

CASE STUDY – GENERATED SAS CODE

AE DOMAIN

- Generated auto-AE.sas – support multiple raw dataset merge to one SDTM domain

```
/*****   START OF AUTO PROGRAM   *****/


/* DATASET: AE_AE */
data AE_AE_TMP;
    length AEACN_SDTM AECONTRT_SDTM AEENDTC AEENRTPT AEENTPT AEOUT_SDTM AEREL_SDTM AESER_SDTM AESEV AESPID AESTDTC $200;
    set RAW.AE;
    STUDYID = strip(STUDY);
    DOMAIN = "AE";
    USUBJID = CATS(STUDY,"/",SUBJECT);
    AESPID = CATS('AE-',AENO);
    AETERM_SDTM = strip(AETERM);
    AELLT = strip(LLT_NAME);
    AELLTCD = LLT_CODE;
    AEDECOD = strip(PT_NAME);
    AEPTCD = PT_CODE;
    AEHLT = strip(HLT_NAME);
    AEHLTCD = HLT_CODE;
    AEHLGT = strip(HLGTNAME);
    AEHLGTCD = HLGTCODE;
    AEBODSYS = strip(SOC_NAME);
    AEBDSYCD = SOC_CODE;
    AESOC = strip(SOC_NAME);
    AESOCCD = SOC_CODE;
    AESEV = upcase(strip(put(AESEVMAX, $AESEV.)));
    AESER_SDTM = upcase(strip(put(AESER, $NY.)));
    AEACN_SDTM = ifc(AEC01AC ne '' , 'MULTIPLE' , ifc(AEACN = '4' , 'DRUG WITHDRAWN' , upcase(strip(put(AEACN, $AEACN.)))));
    AEREL_SDTM = ifc(AEREL = 'OUT9488' , 'REASONABLE POSSIBILITY AE CAUSED BY IP' , ifc(AEREL = 'OUT9487' , 'UNLIKELY AE CAUSED BY IP' , ''));
    AEOUT_SDTM = upcase(strip(put(AEOUT, $OUT.)));
    AECONTRT_SDTM = strip(put(AECONTRT, $NY.));
    %dtc(INVAR_DAT=AESTDAT,INVAR_TIM=AESTTIM,OUTVAR_DTC=AESTDTC);
    %dtc(INVAR_DAT=AEENDAT,INVAR_TIM=AEENTIM,OUTVAR_DTC=AEENDTC);
    AEENRTPT = ifc(AEENDAT = '' and AEOUT IN ('OUT9498','OUT9495') , 'BEFORE' , ifc(AEOUT = 'OUT8275' , 'COINCIDENT' , ifc(AEOUT IN ('OUT9496','OUT9494') , 'ONGOING' , '')));
    AEENTPT = ifc(AEENDAT ne '' , '' , ifc(AEOUT IN ('OUT9498','OUT9496','OUT9495','OUT9494') , 'STUDY TERMINATION' , ifc(AEOUT = 'OUT8275' , 'DEATH' , '')));
    if AETERM = '' then delete;
run;
data AE_AE;
    set AE_AE_TMP(rename=(AEACN=AEACN_RAW  AECONTRT=AECONTRT_RAW  AEOUT=AEOUT_RAW  AEREL=AEREL_RAW  AESER=AESER_RAW  AETERM=AETERM_RAW));
    rename AEACN_SDTM=AEACN  AECONTRT_SDTM=AECONTRT  AEOUT_SDTM=AEOUT  AEREL_SDTM=AEREL  AESER_SDTM=AESER  AETERM_SDTM=AETERM;
run;
proc delete data=AE_AE_TMP; run;

/* DATASET: AE_SERAE */
data AE_SERAE_TMP;
    length AESCONG_SDTM AESDISAB_SDTM AESDTH_SDTM AESHOSP_SDTM AESLIFE_SDTM AESMIE_SDTM AESPID SAEDTC SAEHODTC SAEIADTC $200;
    set RAW.SERAE;
    STUDYID = strip(STUDY);
    USUBJID = CATS(STUDY,"/",SUBJECT);
```

```
proc sort data=AE_AE; by usubjid aeno; run;
proc sort data=AE_SERAE; by usubjid aeno; run;
data AE;
    merge AE_AE AE_SERAE;
    by usubjid aeno;
run;

/*****   END OF AUTO PROGRAM   *****/
```

VS DOMAIN

- Generated auto-VS.sas – supporting sub-block to handle 1:m record mapping

```
*******************************************************************************/
/*****   START OF AUTO PROGRAM   *****/


/* DATASET: VS_VS */
data VS_VS;
    length VISITDTC VSDTC VSORRES VSORRESU VSSTRESC VSSTRESU VSTESTCD VSTPT $200;
    set RAW.VS;
    STUDYID = strip(STUDY);
    DOMAIN = "VS";
    USUBJID = CATS(STUDY,"/",SUBJECT);
    VISITNUM = VISIT;
    %dtc(INVAR_DAT=VSDAT,INVAR_TIM=VSTIM,OUTVAR_DTC=VSDTC);
    VSTPT = strip(PROTSCHD);
    MODULE = ifc(MODULE ne '',MODULE, 'VS');
    %dtc(INVAR_DAT=VIS_DAT,OUTVAR_DTC=VISITDTC);
    IF not missing(SBP) THEN DO;
        VSTESTCD = "SYSBP";
        VSTEST = "Systolic Blood Pressure";
        VSCAT = "PULSE AND BLOOD PRESSURE";
        VSPOS = "UNSPECIFIED";
        VSORRES = strip(put(SBP, best.));
        VSORRESU = "mmHg";
        VSSTRESC = strip(put(SBP, best.));
        VSSTRESN = SBP;
        VSSTRESU = "mmHg";
        OUTPUT;
    END;
    call missing(VSTEST, VSCAT, VSPOS, VSORRES, VSORRESU, VSSTRESC, VSSTRESN, VSSTRESU);

    IF not missing(DBP) THEN DO;
        VSTESTCD = "DIABP";
        VSTEST = "Diastolic Blood Pressure";
        VSCAT = "PULSE AND BLOOD PRESSURE";
        VSPOS = "UNSPECIFIED";
        VSORRES = strip(put(DBP, best.));
        VSORRESU = "mmHg";
        VSSTRESC = strip(put(DBP, best.));
        VSSTRESU = "mmHg";
        OUTPUT;
    END;
    call missing(VSTEST, VSCAT, VSPOS, VSORRES, VSORRESU, VSSTRESC, VSSTRESU);

    IF not missing(WEIGHT) THEN DO;
        VSTESTCD = "WEIGHT";
        VSTEST = "WEIGHT";
        VSCAT = "BODY MEASUREMENT";
        VSORRES = strip(put(WEIGHT, best.));
        VSORRESU = strip(put(WEIGHTU, $VSRES2F.));
```

FUTURE WORK

This Auto-SDTM software is the initial step toward automating clinical data processing. Planned improvements include:

- Integrating with define.xml generation tools
- Integrating with aCRF generation tools
- Enhancing AI-based mapping processes
- Incorporating AI models or agents

CONCLUSION

The Auto-SDTM mapping software provides a scalable, compliant solution for SDTM creation. Combining metadata-driven rules with executable SAS/R generation reduces manual effort, strengthens compliance, and improves efficiency across development programs.

REFERENCES

- A Practical Approach to Automating SDTM Using a Metadata-Driven Method (PHUSE 2024, Paper SM10) by Bob Cui. etc.
- SDTM Specification Automation Using an ML Approach to Provide Mappings (PHUSE 2025, Paper ML20) by Benzi Alex Mathew, Jaya Simha Inampudi

ACKNOWLEDGMENTS

We would like to express our sincere gratitude to the **ClinChoice** Rule-Based Auto-SDTM Development Team for their exceptional technical contributions to this project.
Special thanks go to the BIOM-Programming team—**Zitong Wang, Yunsheng Wang, Tina Wu and Carrie Zhang**—for their expertise in business design and mapping logic, the Data Science team—**Yu Du, John Lin, Tina Ma and Bill Zhang**—for their innovation in streamlining the automation framework, as well as UAT/QC team. Their collaborative efforts in developing this metadata-driven solution have significantly enhanced the efficiency and accuracy of our SDTM standardization process.

CONTACT INFORMATION

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

Author Name: Carrie Zhang
Company: ClinChoice Inc.
Email: carrie.zhang@clinchoice.com

Author Name: Bill Zhang
Company: ClinChoice Inc.
Email: bill.zhang@clinchoice.com