phuse·share
connect·share·advance

Single Day Event

Better CDISC Standards with Metadata Programming

Sunil Gupta
CDISC SME, Consultant and Author

Abhishek Dabral
Director, Clinical Programming

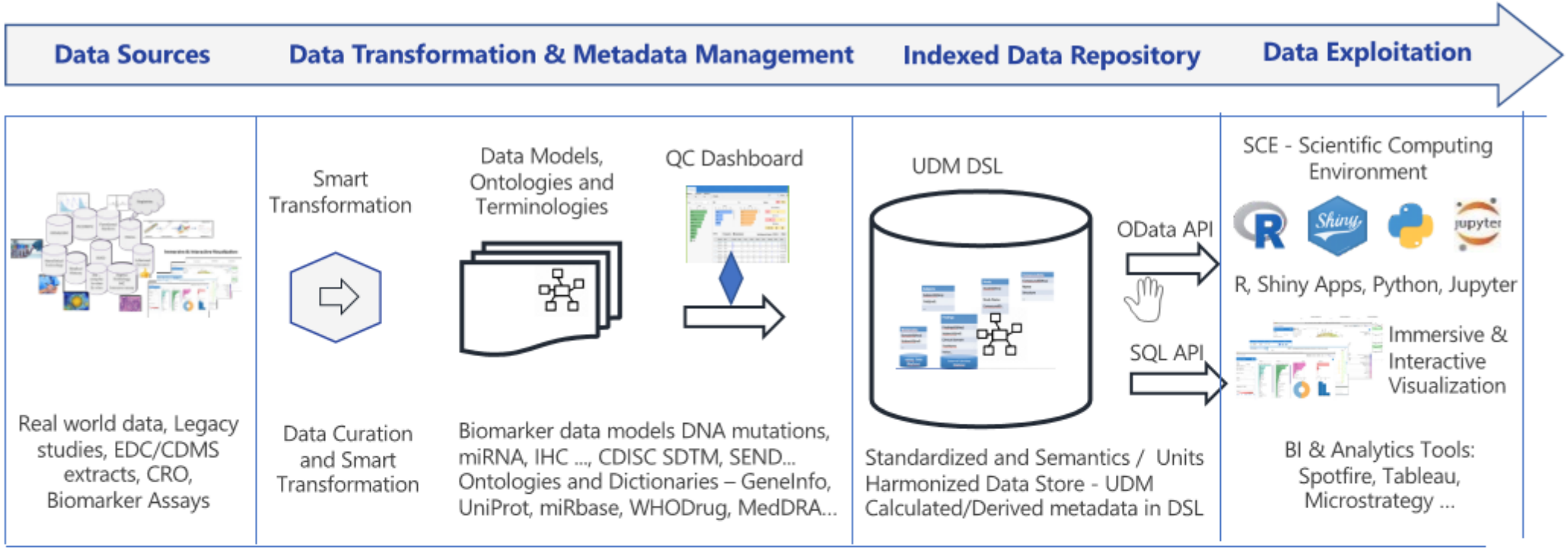
*"Apply the 80/20 rule to ensure the Project **automates 80% of the end-to-end metadata and data processing** needed to generate study artifacts suitable for a regulatory submission." Peter Van Reusel, Sam Hume, CDISC-360 Mission*

phuse.global

Big Data Value Chain

Data Acquisition	Data Analysis	Data Curation	Data Storage	Data Usage
• Structured data • Unstructured data • Event processing • Sensor networks • Protocols • Real-time • Data streams • Multimodality	• Stream mining • Semantic analysis • Machine learning • Information extraction • Linked Data • Data discovery • 'Whole world' semantics • Ecosystems • Community data analysis • Cross-sectorial data analysis	• Data Quality • Trust / Provenance • Annotation • Data validation • Human-Data Interaction • Top-down/Bottom-up • Community / Crowd • Human Computation • Curation at scale • Incentivisation • Automation • Interoperability	• In-Memory DBs • NoSQL DBs • NewSQL DBs • Cloud storage • Query Interfaces • Scalability and Performance • Data Models • Consistency, Availability, Partition-tolerance • Security and Privacy • Standardization	• Decision support • Prediction • In-use analytics • Simulation • Exploration • Visualisation • Modeling • Control • Domain-specific usage

Data Curation is the Repetitive Process to Optimize Data and Metadata to ensure Valuable use of Data.



CDISC 360 MISSION:

SDTM Design And Automation

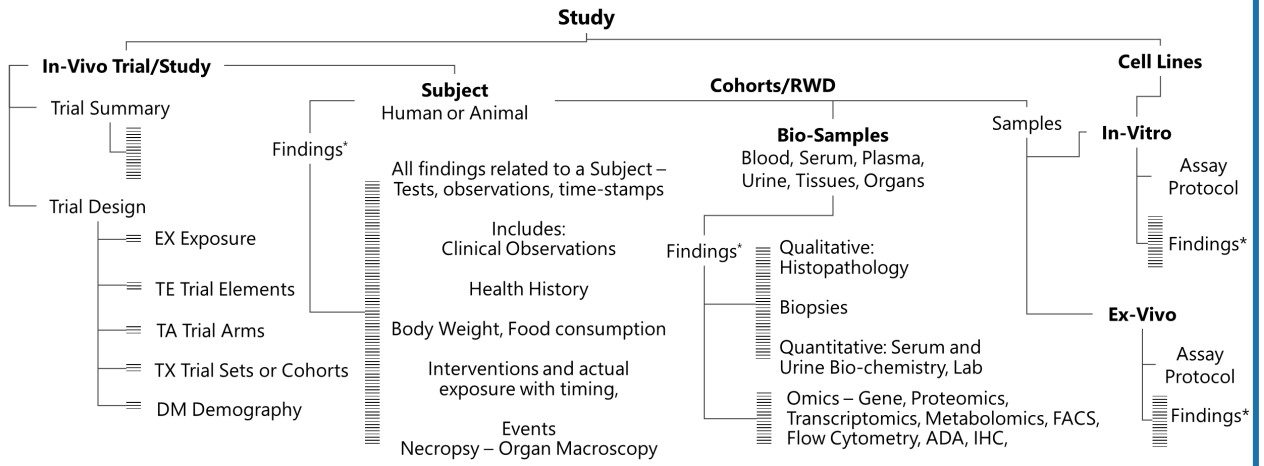
Input: Metadata & Raw Clinical Data

Create End-to-Start Specification

❑ Produce a standards-based, machine readable specification

Generate Start-to-End Metadata

❑ Use standards specification to generate study metadata artifacts
❑ Demonstrate the ability to generate study metadata given a specification

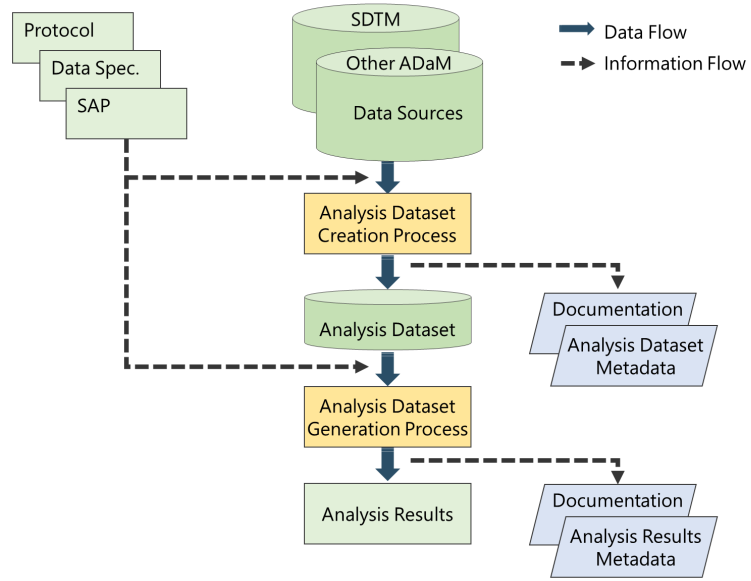


** Findings are held in a common storage model with Observation/Testname, date-Time stamp, Units and other attributes*

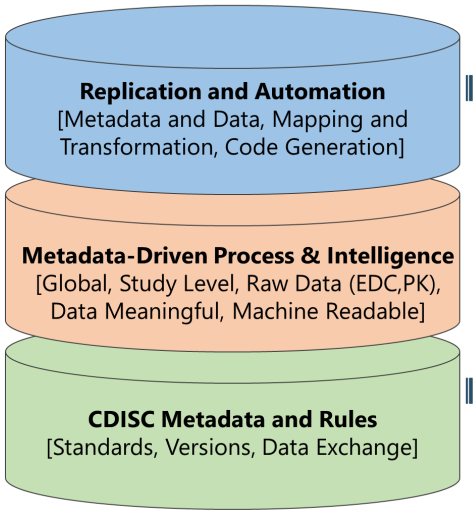
Output: SDTMs, ADaMs, Define.xml & TFLs

Data Curation

❑ Repetitive Process to Optimize Data and Metadata to ensure Valuable use of Data



Input



Transformation and Automation: Reusability & Repeatability

Output

Transformation Data Start-to-End

❑ Use machine-readable metadata to generate study data artifacts
❑ Demonstrate the ability execute data transformations given the study

Replication and automation are the focuses

- Use or create utilities to replicate the process: Project Set Up, Mapping Specification, Mapping Creation
- Use analytics tool to identify the areas for replication and automation: Data Profiling & Data Rules for Source Data Review / Edit Checks

Metadata-driven process is the key for automation

- Metadata makes data meaningful
- Metadata is machine readable
- Metadata is the base for automation

Standard adoption is the key for code reusability

- Train people to understand the standards
- Define standard templates
- Build public libraries for code snippets and public transformation: Custom functions, procedures and packages; public data rules; and public Experts
- Group code snippets and functional transformation into modular mapping and transformation: pluggable maps
- Define workflow to govern the process: Workflow Manager and Process Flows

End-to-End Clinical Study MetaData-Driven Process and Intelligence CDISC and Submission Flow

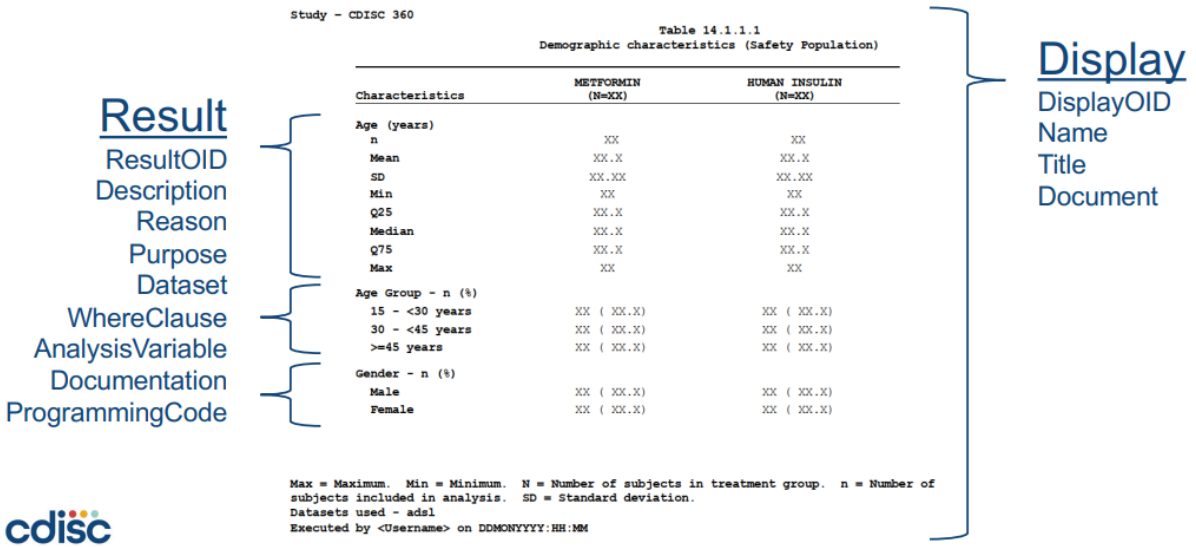
Raw Data, Metadata, Xbiom Tool → CDISC / Analysis → Documentation → QA

EDC / Labs / CRF	Metadata / CDISC Deliverables			Study Documentation Define.xml	Regulatory Compliance eDV / SDRG / ADRG
	SDTMs	ADAMs	TFLs		
DATA: Raw Codelists	Standard Domains Standard Variables Standard Terminology Codelists	Safety / Efficacy Derived Variables Codelists	SAP	Documentation Control Terminology Value-Level Metadata Raw / Derived Variables	Documentation Data Issues Compliance Issues
METADATA / **CDASH** SPECIFICATIONS: Attributes, Structure, **PRM**	**SDTM IG Rules** **Control Term IG Rules** **MedDRA** Export Specifications	**ADAM IG Rules** **Control Term IG Rules** (Optional) Export Specifications	**ARMs** BDS Independent of ADaMs	Define.xml IG Rules SDTMs / ADaMs Snapshot Integrated Links to CRF pages User-Interface Edits	Snapshots / Links
USER INTERFACE MACHINE LEARNING PRODUCTIVITY:	Joins / Transpose Auto / User Mapping Templates Drop-down lists	SAP Mapping Auto / User Mapping SAP Cohorts Drop-down lists	SAP Cohorts Domain Templates Drop-Down Lists	IG Mapping Templates	Template Mapping PhUSE Templates
TRADITIONAL PROGRAMMING PRODUCTIVITY:	Source / QC Attribute Macros Variable Macros	Attribute Macros Variable Macros	Reporting Macros	Separate Tool Out-of-Sync	Separate Tool Manual Updates

What are ‘Best Practices’ Applications of Metadata Programming?

- Dataset Specifications from Excel file
- Proc SQL Dictionary Tables to get Dataset Attributes
- Cross-referencing Datasets
- Defensive Programming for Variable Type Specific Syntax
- Identifying Special Characters
- Standardizing Raw Data Values
- Loop through one macro variable with a list of values
- Standardize derivation of ISODATE (--DTC) variables
- Dynamically Executing SAS code
- System Environment Clean-up

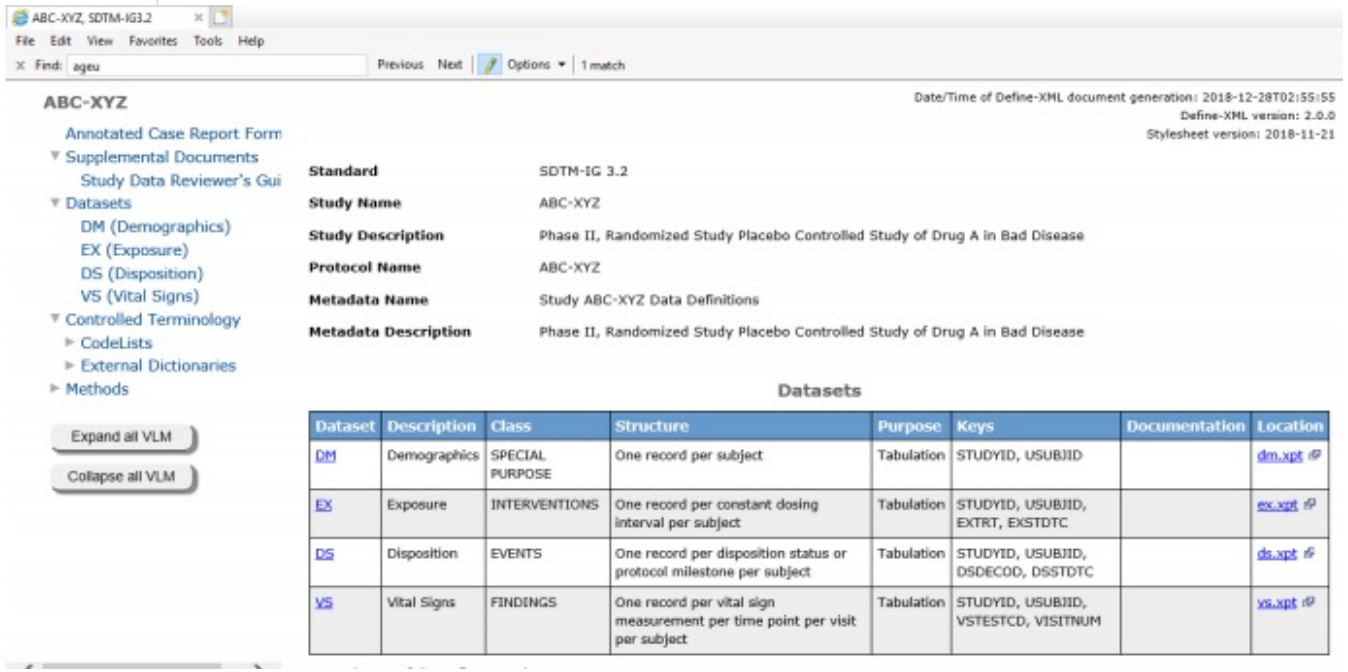
CDISC ARM v1 Metadata



Define (.xls) Specification File / (.xml) Metadata File

	A	B	C	D	E	F	G	
1	Order	Dataset	Variable	Label	Data Type	Length	Significant Digit	Forr
2	1	AE	STUDYID	Study Identifier	text	8		
3	2	AE	DOMAIN	Domain Abbreviation	text	2		
4	3	AE	USUBJID	Unique Subject Identifier	text	17		
5	6	AE	AESEQ	Sequence Number	float	8	0	
6	9	AE	AESPID	Sponsor-Defined Identifier	text	29		
7	12	AE	AETERM	Reported Term for the Adve	text	23		
8	14	AE	AELLT	Lowest Level Term	text	1		
9	15	AE	AELLTCD	Lowest Level Term Code	integer	8		
10	16	AE	AEDECOD	Dictionary-Derived Term	text	1		
11	17	AE	AEPTCD	Preferred Term Code	integer	8		
12	18	AE	AEHLT	High Level Term	text	1		
13	19	AE	AEHLTCD	High Level Term Code	integer	8		
14	20	AE	AEHLGT	High Level Group Term	text	1		
15	21	AE	AEHLGTCD	High Level Group Term Cod	integer	8		
16	28	AE	AEBODSYS	Body System or Organ Class	text	1		
17	29	AE	AEBDSYCD	Body System or Organ Class	integer	8		
18	30	AE	AESOC	Primary System Organ Class	text	1		
19	31	AE	AESOCCD	Primary System Organ Class	integer	8		
20	38	AE	AESEV	Severity/Intensity	text	28		
21	39	AE	AESER	Serious Event	text	1		
22	40	AE	AEACN	Action Taken with Study Tre	text	32		
23	41	AE	AEACNOTH	Other Action Taken	text	19		
24	43	AE	AEREL	Causality	text	11		
25	46	AE	AEOUT	Outcome of Adverse Event	text	53		

Study | Datasets | Variables | ValueLevel | WhereClauses | Codelists | Dictionaries



Metadata Programming Concepts – Loops, Conditions and Syntax

Loop Across Variables

Loop Across Records

	A	B	C	D	E	F
1	DOMAI	VARNU	VARIABLE	TYPE	LENG	LABEL
2	AE	1	STUDYID	text	15	Study Identifier
3	AE	2	DOMAIN	text	2	Domain Abbreviation
4	AE	3	USUBJID	text	25	Unique Subject Identifier
5	AE	4	AESEQ	integer	8	Sequence Number
6	AE	5	AETERM	text	200	Reported Term for the Adverse Event
7	AE	6	AEDECOD	text	200	Dictionary-Derived Term
8	AE	7	AEBODSYS	text	200	Body System or Organ Class
9	AE	8	AESEV	text	40	Severity/Intensity
10	AE	9	AESER	text	40	Serious Event
11	AE	10	AEACN	text	40	Action Taken with Study Treatment
12	AE	11	AEREL	text	40	Causality
13	AE	12	AESTDTC	date	16	Start Date/Time of Adverse Event
14	AE	13	AEENDTC	date	16	End Date/Time of Adverse Event
15	AE	14	AESTDY	integer	8	Study Day of Start of Adverse Event
16	AE	15	AEENDY	integer	8	Study Day of End of Adverse Event

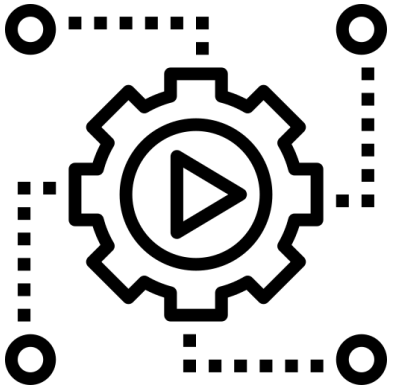
SAS Code Generator

```
attrib studyid length=15 label='Study Identifier';
aehltcd = input(hltc_char, best32.);
%report(var=Age , type=num);
```

Metadata Process

- Create macro variables by variables & rows
- Loop across records to Build SAS Statements
- Loop across records to Build SAS Procedures
- Directly apply conditions and macro calls
- Expects clean and valid metadata
- Complement Dictionary Tables
- Indirect macro variable references
- Proc SQL cross-reference joins and excepts
- Inventory of Libraries and Files
- Build SAS Code before writing to SAS Program

SDTM Generator



‘Data-Driven’ Metadata Programming are Best Practices

Rules: SDTM Attributes Specifications

	A	B	C	D	E	F
1	DOMAI	VARNU	VARIABLE	TYPE	LENG	LABEL
2	AE	1	STUDYID	text	15	Study Identifier
3	AE	2	DOMAIN	text	2	Domain Abbreviation
4	AE	3	USUBJID	text	25	Unique Subject Identifier
5	AE	4	AESEQ	integer	8	Sequence Number
6	AE	5	AETERM	text	200	Reported Term for the Adverse Event
7	AE	6	AEDECOD	text	200	Dictionary-Derived Term
8	AE	7	AEBODSYS	text	200	Body System or Organ Class
9	AE	8	AESEV	text	40	Severity/Intensity
10	AE	9	AESER	text	40	Serious Event
11	AE	10	AEACN	text	40	Action Taken with Study Treatment
12	AE	11	AEREL	text	40	Causality
13	AE	12	AESTDTC	date	16	Start Date/Time of Adverse Event
14	AE	13	AEENDTC	date	16	End Date/Time of Adverse Event
15	AE	14	AESTDY	integer	8	Study Day of Start of Adverse Event
16	AE	15	AEENDY	integer	8	Study Day of End of Adverse Event

Process: SDTM Mapping Specifications

Obs	Domain	Variable	Source_ domain	Source_var	mapping	group
1	AE	USUBJID	AE	subjid_raw_char	USUBJID = subjid_raw_char;	1
2	AE	AEHLTCD	AE	hltc_num	AEHLTCD = hltc_num;	1
3	AE	USUBJID	AECD	subjid_raw_char	USUBJID = subjid_raw_char;	2
4	AE	USUBJID	SFAE	subjid_raw_char	USUBJID = subjid_raw_char;	3
5	AE	AEHLTCD	SFAE	hltc_char	AEHLTCD = input(hltc_char,best32.);	3
6	AE	USUBJID	SFAECD	subjid_raw_char	USUBJID = subjid_raw_char;	4
7	AE	USUBJID	SFAESS	subjid_raw_char	USUBJID = subjid_raw_char;	5
8	AE	AEHLTCD	SFAESS	hltc_char	AEHLTCD = input(hltc_char,hltc_c_ i.);	5

Table 4.2: SAS data set of Excel file converted to data set mappings.

Transform to create SDTM Shells

Transform to standardize Raw Data

One Time Design Setup, Loop through SDTMs, Variables and Attributes,
Code Generator, No Manual Coding, Faster SDTM Cycles

Higher Level Design and Macro Programming - Snipit

Rules: SDTM Attributes Specifications

****** Read in Excel file/Dataset of specs ****;**

****** Check spec variable length content type and value ****;**

****** Generate domain macro variables with suffix for each record in spec file – varnam#, vartyp#, varlen#, varlbl# ****;**

```
data _null_;
 set specin  end=eof nobs=numb;
   call symput ('varnam'||trim(left(put(_n_,3.))), trim(variable));
   call symput ('vartyp'||trim(left(put(_n_,3.))), trim(type));
   call symput ('varlen'||trim(left(put(_n_,3.))), trim(left(length)));
   call symput ('varlbl'||trim(left(put(_n_,3.))),trim(label));
   if eof then call symput ('attrnum', trim(left(put(numb, 3.))));
run;
```

Loop through all records and macro variables to create attribute statements;

```
 %do i=1 %to &attrnum. ;
   %if %upcase(&&vartyp&i..) = CHAR %then %do;
     %if &&varlen&i.. ne %then %do;
       attrib &&varnam&i.. length=$&&varlen&i..
              label="&&varlbl&i..";
               &&varnam&i = '';
     %end;
     %else %do;
       attrib &&varnam&i.. length=$125 label="&&varlbl&i..";
              &&varnam&i = '';
     %end;
   %end;
```

Higher Level Design and Macro Programming - Snipit

Process: SDTM Mapping Specifications

```
%*** Loop through each extract to bring in the data. ***;
%do &__z_i= 1 %to &number_extracts;

 %*** Derive source data, rename variables, variable to the identify the input source data ***;
 %*** Derive copy variables, variables format transformations, drop variables ***;

%*** Extract ***;
 data __z_i_source_der_&__z_i;
  attrib &__z_i_source_dom_var_name length=$50 label='source domain';
 %*** Derive source data and rename variables ***;
 set &inlib..&__z_i_source_name_in.&__z_i_source_var_rename;

%*** Identify the input source data name ***;
 &__z_i_source_dom_var_name = "&__z_i_source_name_in";

 %*** copy variables ***;
 &__z_i_source_var_copy_syntax;

 %*** variables format transformations ***;
 &__z_i_derive_var_syntax;

 %*** drop variables ***;
 &__z_i_drop_var_syntax;
 run;
%end;
```

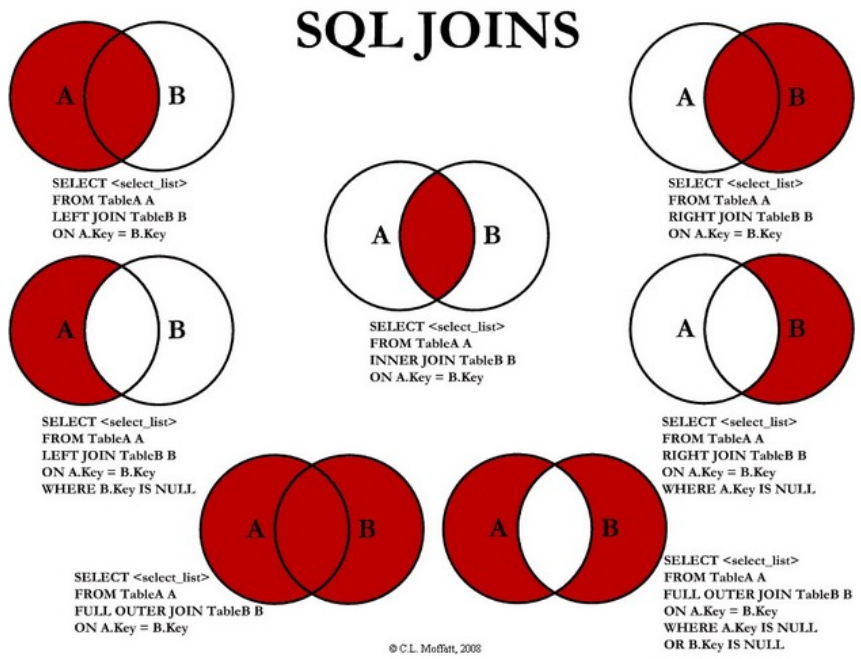
```
proc format;
 value $sex 'F' = 'Female'
            'M' = 'Male';
run

data dm;
 set demog;
 sex = put(raw_sex, $sex.);
run;
```

Use Dictionary Tables to Access Metadata Files in Programs

Table 1. Organized List of DICTIONARY Tables and SASHELP Views

Category	DESCRIPTION	DICTIONARY Table	SASHELP VIEWS*
SAS Session	SAS Options	OPTIONS**	VOPTION
	SAS/Graph Options	GOPTIONS**	VGOPT
	TITLE Statements	TITLES	VTITLE
	Defined macro variables	MACROS	VMACRO
	LIBNAME Information	LIBNAMES	VLIBNAM
	Available Engines	ENGINES	VENGINE
	Files defined in FILENAME statements, or implicitly	EXTFILES	VEXTFL
Members	Tables, Catalogs, and Views	MEMBERS	VMEMBER
Data Set and Related Metadata	Table and Table-Specific Information	TABLES	VTABLE
	View and View-Specific Information	VIEWS	VVIEW
	Columns from every table	COLUMNS	VCOLUMN
	DICTIONARY tables and their columns	DICTIONARIES	VDCTNRY
	Indexes	INDEXES	VINDEX
Catalogs and Related Metadata	Catalog and Catalog-Specific Information	CATALOGS	VCATALG
	Available Formats	FORMATS***	VFORMAT
	Styles	STYLES	VSTYLE
Constraints	Check Constraints	CHECK_CONSTRAINTS	VCHKCON
	Referential Constraints	REFERENTIAL_CONSTRAINTS	VREFCON
	Table Constraints	TABLE_CONSTRAINTS	VTABCON
	Constraint table usage	CONSTRAINT_TABLE_USAGE	VCNTABU
	Constraint column usage	CONSTRAINT_COLUMN_USAGE	VCNCOLU
Other	Remember information	REMEMBER****	VREMEMB

```
PROC SQL;
 CREATE TABLE RAW_VARS AS
 SELECT UNIQUE libname, memname,
name, type, length, label,
format
 FROM SASHELP.VCOLUMN
 WHERE upcase(libname) = 'RAW';
QUIT;
```



Defense is the Best Offence – Plan for the Unexpected

Up to 70% of Macro Development may be on Data Screening and Confirming Assumptions
Tools are only as Effective as they are being used Frequently and Correctly

Leverage Metadata Functions

Dictionary/Array Reference Functions	Result See SAS Paper, SAS Paper 2
CALL LABEL ROUTINE	Assign label to variable
VLABEL()	Returns the variable's label Example, Y=VLABEL(X); or Y=VARLABEL(X);
VLENGTH()	Returns the length of the string or numeric value from a character or numeric variable
VNAME()	Returns the variable's name, argument is often arrary index reference See also INDEX("&VAR", 'END') > 0 to compare variable names
VTYPE()	Returns the variable's type (N, C)
VVALUE()	Returns the formatted value

Leverage SAS SCL Functions

Function name	Function action, Example
EXIST	Verifies the existence of a SAS member (dataset, catalog), returning a 1 or 0: `%if %sysfunc(exist(work.sae)) %then…`
OPEN	Opens a SAS dataset and returns a value (id). Many of the subsequent functions in this table use the id as an argument (and not the dataset name): `%let dsid = %sysfunc(open(crt.d_ae));`
CLOSE	Closes the dataset given by the id, and returns a value (0 if successful). Any dataset opened with the OPEN function should be closed with the CLOSE function: `%let rc = %sysfunc(close(&dsid));`
DSNAME	Returns the dataset name associated with a dataset id: `%let dsname = %sysfunc(dsname(&dsid));`
ATTRC	Returns the value of a character attribute for the dataset associated with the id: `%let att = %sysfunc(attrc(&dsid,attrib));` where *attrib* is (amongst others) sortedby - sort order if dataset is sorted (otherwise blank) label - dataset label mem - dataset name
ATTRN	Returns the value of a numeric attribute for the dataset associated with the id: `%let att = %sysfunc(attrn(&dsid,attrib));` where *attrib* is (amongst others) nobs - number of observations nvars - number of variables crdte - date created (SAS date time format)

Know Your Data: Be Aware of Special Characters

Macro quoting function nrstr() keeps from resolving &thisds (masks the &)

Remove non-printable characters such as carriage returns.

```
comment = compress(strip(Comment), , 'kw');
```

Remove carriage return (‘OD’x) and line feed (‘OA’x) hidden characters.

```
comment = compress(comment, ’0D0A’x);
```

Leverage Variable Type and Content Functions

ANYALNUM	Search for alphanumeric characters
ANYALPHA	Search for alphabetical characters
ANYCNTRL	Search for control characters
ANYDIGIT	Search for digit characters
ANYFIRST	Search for characters that are valid for being the first character of a SAS variable name
ANYGRAPH	Search for graphic characters
ANYLOWER	Search for lowercase letters
ANYNAME	Search for characters that are valid in a SAS variable name
ANYPRINT	Search for printable characters
ANYPUNCT	Search for punctuation characters
ANYSPACE	Search for white-space character, which include blank, horizontal and vertical tab, carriage return, line feed, and form feed
ANYUPPER	Search for uppercase letters
ANYXDIGIT	Search for hexadecimal characters

Create Custom Functions Proc FCMP: Derivation of ISO Dates (--DTC)

```
/*-----------------------------------------------------------------
Description
1) dtc_from_dttm: FCMP function to derive DTC variable from date and time
variables
2) dtc_from_dt: FCMP function to derive DTC variable from date variable
3) day_from_dtc: FCMP function to derive Study Day from Date part of DTC
variable
 -----------------------------------------------------------------*/
```

Define Input and Output Parameters, Apply Program Logic, Utilize Across Studies Similar to SAS Procedures

```
proc fcmp outlib=work.funcs.dtc_from_dttm;
 function dtc_from_dttm( datvar$, timvar$ ) $;
 attrib _return_value1 length=$50;
 _return_value1 = '';
 if not missing(datvar) then do;
  if compress(scan(upcase(datvar),1,'/')) in ( '' 'UNK')  then
  _return_value1 = '-';
```

Dynamically Execute SAS Code

1) 'Driver' Dataset

```
proc sql;
 create table work.Vars as
 select name, type
 from dictionary.columns
 where memname="CLASS" and
libname="SASHELP";
quit;
```

2) SAS Macro Called by Variable

```
%macro report(var= , type= );
 %if &type=char %then %do;
 proc freq data=sashelp.class;
  table &var;
 run;
 %end;
 %else %do;
 proc means data=sashelp.class;
  var &var;
 run;
 %end;
%mend report;
```

3) Data _null_ and Call Execute

```
data _null_;
 set work.Vars;
 call
execute('%report(var='||strip(name)||' ,
type='||strip(type)||');');
run;
```

4) SAS Macro Code Generator

```
%report(var=Age, type=num);
%report(var=Sex, type=char);
```

Know your System Environment

1. OPTIONS
2. GOPTIONS
3. System macro variables
4. Temporary data sets and formats
5. Libraries
6. Filenames
7. Titles and Footnotes
8. Macros
9. Macro variables

```
title; footnote;
goptions reset=all;
%let syscc = 0; ** Operating environment condition code
**;
%let sysrc = 0; ** Operating system condition code **;
%let syslibrc = 0; ** Libname statement condition code **;
%let sysfilrc = 0; ** Filename statement condition code
**;
proc datasets library = work kill; quit;

proc sql noprint;
 select unique libname into :mylibs separated by ' clear;
libname '
 from dictionary.libnames where libname not in
('MAPS','SASHELP','SASUSER','WORK');
quit;
libname &mylibs clear;

proc sql noprint;
 select name into :mymacrovars separated by ' '
 from dictionary.macros where scope = 'GLOBAL'; quit;
%symdel &mymacrovars mymacrovars;
```

SDTM Curation Enables Monitoring and Accretive Resolution of Data Issues

Clinical Data Issues

- Missing Data Values
- Invalid Dates and Data Values
- Character / Numeric Variable Type Conversion
- Zero Records

Study Protocol Data Issues

- Standardized Terms – no loss of data or context
- Lab Data – identify duplicate records, missing values, invalid units, etc.
- Primary Endpoints – correctly derived
- Survival Analysis – subgroup analysis
- Safety – maximum patients and events
- Deaths – maximum patients
- Related Adverse Events – minimum patients and events
- Protocol Compliance – visit in visit window range

Auto-Mapping and Continuous Learning Process

CDISC 360: Apply the 80/20 rule to ensure the **Project automates 80%** of the end-to-end metadata and data processing needed to generate study artifacts suitable for a regulatory submission.

Overall Process

- Pre-processing Batch
  - **Variable** Mapping Methods
  - Control Terms Mapping Methods
  - User Approval Methods
  - New Variable Derivations
- Data Update Batch

A. Variable Mapping Methods

1. Direct
2. Transformation, SQL, ex. trim, concatenating
3. Transpose to Vertical Structure
4. One Raw Data to Multiple SDTMs
5. Multiple Raw Data to One SDTM

B. Control Terms Mapping Methods

1. Exact Value Match
2. Approximate Value Match

D. 100% User Approval Methods

1. Machine Recommended*
2. Previous Decision**

3. Preview Raw data and SDTM standard values
4. SUPPXX, RELREC, FA

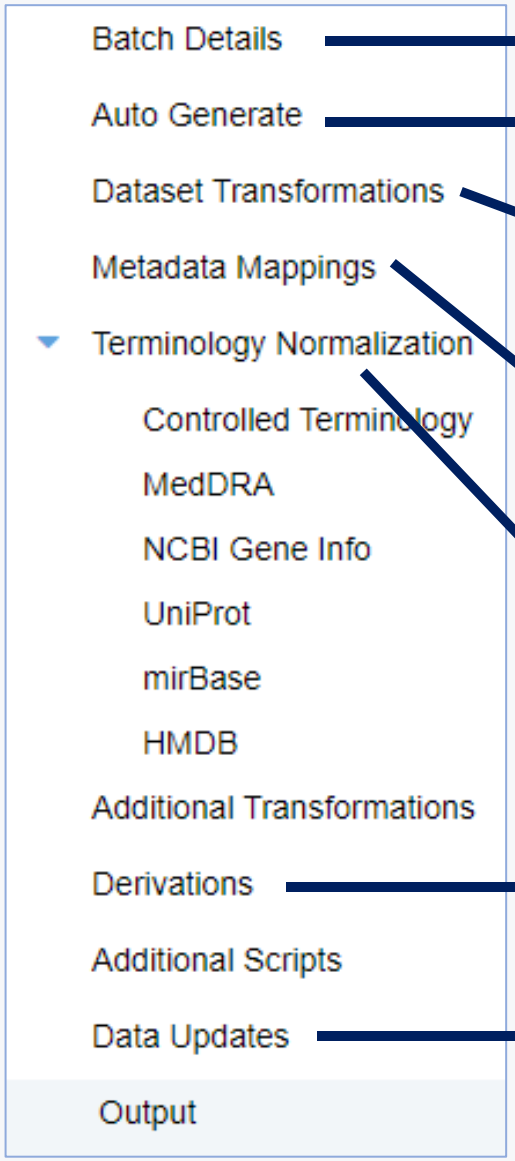
E. New Variable Derivations

DY, STDY, ENDY, DTC, SDTC, ENDTC, BLFL, VISIT

* Learn from sample studies, ** Learn from clinical studies

Xbiom Platform Screenshots

Smart Transformation Module sections Applies the 80/20 Principle



Batch Details: Provide input (source) and output (target model and CT)

Auto Generate: Used if STUDYID, DOMAIN, SEQ (Sequence) values to be automatically generated by system.

Scripts for any Transformations: Three sections (Dataset Transformations, Additional Transformations, Additional Scripts) are provided to write any custom scripts in SQL, Python or PySpark languages for file processing or for any data derivations or corrections.

Metadata Mappings: Used to map source data structure to target model domains and columns. System recommends mappings based on training sets and users' previous decisions. User can approve or modify the recommended mappings.

Terminology Normalization: Used to map the source terms to target terms. External dictionaries like MedDRA, NCBI Gene Info, UniProt, mirBase, HMDB also supported. System recommends mappings to target terms based on Xbiom global CT and loaded external dictionaries. User can approve or modify the recommended mappings.

Derivations: to derives the data, if missed to collect in source systems.

Data Updates: To perform custom data updates.

Speaker: Bhav

Machine-readable Mapping Specifications

Source				Mapping					Target						
Source Sequence	Source Library	Source Dataset	Source Variable	Map Sequence	Origin	Method	Comment	Code List	Target Library	Target Dataset	Target Variable	Target Description	Target Data Type	Target Legnth	Target Sorting Order
1	CDASH	VS			Assigned		CDISC360-2		SDTM	VS	STUDYID	Study Identifier	text	10	1
1	CDASH	VS			Assigned		VS	DOMAIN	SDTM	VS	DOMAIN	Domain Abbreviation	text	2	2
1	CDASH	VS	SUBJID		Assigned	ALL.USUBJID			SDTM	VS	USUBJID	Unique Subject Identifier	text	14	3
1	CDASH	VS			Assigned	VS.VSSPID			SDTM	VS	VSSPID	Sponsor-Defined Identifier	text	4	5
1	CDASH	VS	VISIT		Convert			VISITNUM	SDTM	VS	VISITNUM	Visit Number	integer	8	16
1	CDASH	VS	VISIT		Predecessor			VISIT	SDTM	VS	VISIT	Visit Name	text	18	17
1	CDASH	VS	VSDAT		Assigned	VS.VSDTC			SDTM	VS	VSDTC	Date/Time of Measurements	date	10	19
1	CDASH	VS	VISDAT		Assigned	VS.VSDTC			SDTM	VS	VSDTC	Date/Time of Measurements	date	10	19
1	CDASH	VS			Derived	VS.VSBLFL			SDTM	VS	VSBLFL	Baseline Flag	text	1	14
2	SDTM	DM	RFSTDTC		Derived	VS.VSDY			SDTM	VS	VSDY	Study Day of Vital Signs	integer	8	20
2			VSDTC		Derived	VS.VSDY			SDTM	VS	VSDY	Study Day of Vital Signs	integer	8	20
3	SDTM	SV	VISITDY	1	Predecessor				SDTM	VS	VISITDY	Planned Study Day of Visit	integer	8	18
3	SDTM	SV	EPOCH	2	Predecessor			EPOCH	SDTM	VS	EPOCH	Epoch	text	9	15
4				3	Assigned	VS.VSTESTCD		VSTESTCD	SDTM	VS	VSTESTCD	Vital Signs Test Short Name	text	6	6
4				4	Derived	VS.VSORRES			SDTM	VS	VSORRES	Result or Finding in Original Units	text	4	9
4				5	Derived	VS.VSORRESU		VSUNIT	SDTM	VS	VSORRESU	Original Units	text	9	10
4				6	Assigned	VS.VSSTRESU		VSUNIT	SDTM	VS	VSSTRESU	Standard Units	text	9	13
4				7	Derived	VS.VSSTRESN			SDTM	VS	VSSTRESN	Numeric Result/Finding in Standard Units	float	8	12
4				8	Derived	VS.VSSTRESC			SDTM	VS	VSSTRESC	Character Result/Finding in Std Format	text	4	11
4				9	Assigned	VS.VSPOS		VSPOS	SDTM	VS	VSPOS	Position	text	7	13
5			VSTESTCD		Convert			VSTEST	SDTM	VS	VSTEST	Vital Signs Test Name	text	24	7
5			VSTESTCD		Convert			VSCAT	SDTM	VS	VSCAT	Category for Vital Signs	text	16	8
5					Derived	VS.VSSEQ			SDTM	VS	VSSEQ	Sequence Number	integer	8	4

cdisc

Export SDTM Metadata Mapping Specification for Traceability and Compliance

Excel file can be read by SAS programs to convert Raw data to SDTMs

Raw SDTM Datasets **Variable Derivations** **SDTM Variables**

	A	B	C	D	E	F	G	H
1	Source File Name	Target Domain	Source Column Name	Source Column Lab	Mapping	Target Variable	Status	Parent Column
307	cm	CM	INSTANCENAME		Direct	VISIT	Approved	
324	cm	CM	CMINDC		Direct	CMINDC	Recommended (Previous Decision)	
333	cm	CM	CMONGO_STD		Direct	CMMODIFY	Recommended	
341	cm	CM	CMDOSU		Direct	CMDOSU	Recommended (Previous Decision)	
351	cm	CM	CMTRT		Direct	CMTRT	Recommended (Previous Decision)	
356	cm	CM	CMTRT_ATC2		Direct	CMSCAT	Approved	
360	cm	CM	CMTRT_ATC4		Direct	CMCLAS	Approved	
361	cm	CM	CMTRT_ATC4_CODE		Direct	CMCLASCD	Approved	
372	cm	CM	substring(CMINDC,3,3)		Expression	RVALUE	Approved	CMINDC
373	cm	CM	'SPID'		Expression	RVAR	Approved	CMINDC
374	cm	CM	''		Expression	POOLID	Recommended (Previous Decision)	SITEGROUP
375	cm	CM	case when CMDOSFRQ='Other' then CONCAT_WS(':','Other',CMFRSPEC) else CMDOSFRQ end		Expression	CMDOSFRQ	Approved	CMDOSFRQ
376	cm	CM	case when CMROUTE='Other' then CONCAT_WS(':','Other',CMRTSPEC) else CMROUTE end		Expression	CMROUTE	Approved	CMROUTE
377	cm	CM	case when CMTRT_PRODUCT<>'' then CMTRT_PRODUCT else CMTRT end		Expression	CMDECOD	Approved	CMTRT_PRODUCT

Metadata Mappings | Transpose Mapping | Controlled Terminology | MedDRA | Additional Transformation | Additional Script | Data ...

Metadata Programming is the Solution for SDTM Automation & Compliance

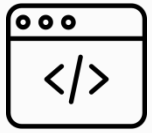
Sunil Gupta
CDISC SME, Trainer & Author

Abhishek Dabral
Director, Clinical Programming

Manage Projects with Automation and Standards

- ✓ One SCE integration tool for all Submission Deliverables
- ✓ Reduce Time and Budget per Clinical Study

Manage Submission Process with Low-Code Programming



- ✓ Reduce writing SAS programs and macros
- ✓ Faster SDTMs, Define.xml and SDRG
- ✓ Auto Generate SDTM Mapping Specifications

Monitor Safety Data Issues with Early Alerts

- ✓ Faster Ingestion, Curation and Harmonization
- ✓ User Interface to create SAP Cohorts

Explore with Pre-defined Templates

- ✓ Reduce Time to Tables, Lists and Figures
- ✓ Drill down from summary to patient level detail

- ❖ CDISC-360 Mission: SDTM Design and Automation (Download PDF)
- ❖ End-to-End Clinical Study MetaData-Driven Process (Download PDF)