USING PROC TEMPLATE TO CONVERT SAS DATA TO DEFINE.XML

Lucius A. Reinbolt, Steven Kirby, Matthew Wiedel, Aleksandra Stein, Vanessa Huang and Nancy Wang

Celerion

DEFINE.XML OVERVIEW

Well formed data contains all information needed to understand study results, but that information is typically not easily accessible to end users; combining the data with xml format data documentation gives end-users a data and documentation package that is complete, user friendly and ready to submit to the FDA. Define.xml data documentation has internal and external links that allow end users to quickly find the level of information they need, from source CRF references to variable lists to code lists and comments associated with key content.

TWO-STEP DATA-DRIVEN DEFINE.XML CREATION USING PROC TEMPLATE

A two-step data-driven process is used to create the define.xml document. The first step is the production of six SAS © format define data sets by mining the data for unique information. The six define data sets describe the data at many levels and are optimized for use as inputs in downstream processing. The next step is integration of the SAS® define data into an xml formatted document through a robust approach driven by SAS© PROC TEMPLATE. Sorted define data sets are restructured and transformed into a group of tagsets that comprise the define document. The benefit of this approach is the concise and powerful application of PROC TEMPLATE that presents the define data through a series of define events triggered by the six levels of metadata. The PROC TEMPLATE code is simple to alter and run, allowing more time to focus on the data itself.

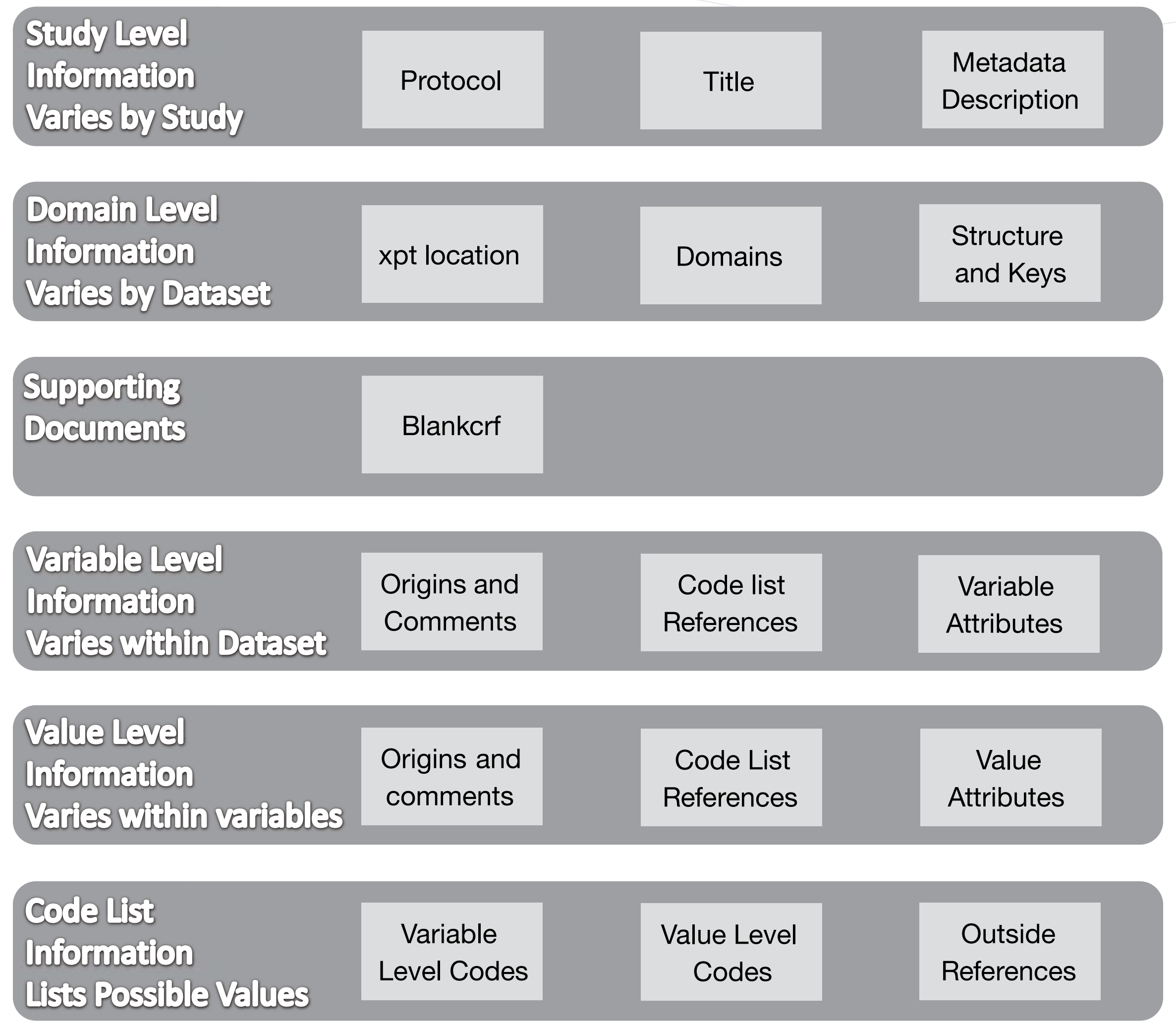
DEFINE DATA GENERATION USING DATA MINING

Simple iterative looping through each submission domain is used to mine the data for the information used to create the define.xml. The data are mined for all unique content at increasingly specific levels; that information is output as six SAS data sets. There is some flexibility in how the datasets are formed, but it is important to use variable characteristics that are consistent with the xsl stylesheet (define-0-0.xsl) used to support define.xml presentation. Supplemental metadata are used to provide a limited amount of additional content. Examples of content added or adjusted through supplemental metadata are variable and value origins, derivation comments, pre-printed CRF code lists, and code list names. That supplemental metadata is typically available from the electronic data mapping specifications used to drive the SDTM, ADaM or custom mapping. With data characteristics driven by the data, the risk of a non-representative define.xml is reduced. Code list names and attributes from the variable (unique result values by variable) and value (unique results within a variable and within a variable value) levels are used to trigger generation of code lists, building in consistency and allowing the programmer to easily control what lists are presented.

This poster has been modified from the original version presented to adhere to Celerion brand guidelines.

DEFINE.XML LEVELS OF INFORMATION

XML format allows for multiple levels of information to be gracefully linked together. The Diagram below shows the different levels of information. The data (and some supplementary metadata) are mined to generate six SAS data sets that correspond with these levels of information. That information is then integrated into to form the define.xml using PROC TEMPLATE.



Study Level Information Varies by Study: Protocol | Title | Metadata Description

Domain Level Information Varies by Dataset: xpt location | Domains | Structure and Keys

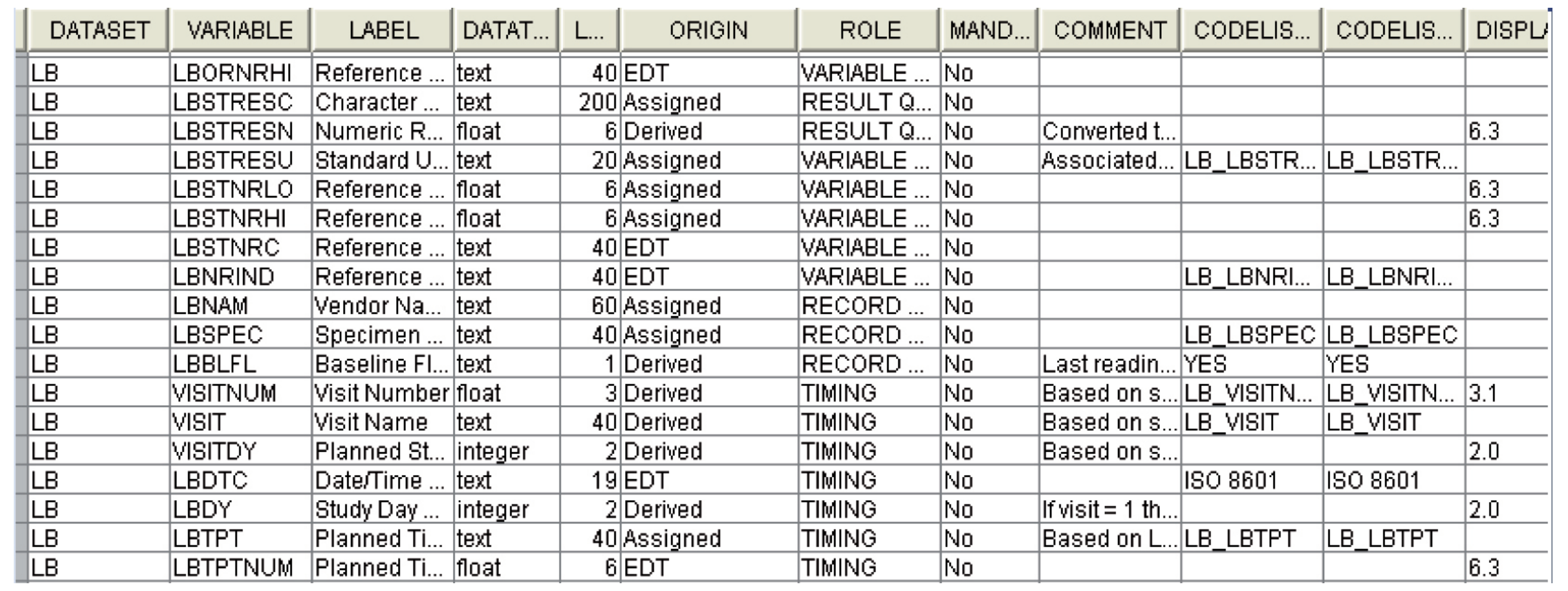
Supporting Documents: Blankcrf

Variable Level Information Varies within Dataset: Origins and Comments | Code list References | Variable Attributes

Value Level Information Varies within variables: Origins and comments | Code List References | Value Attributes

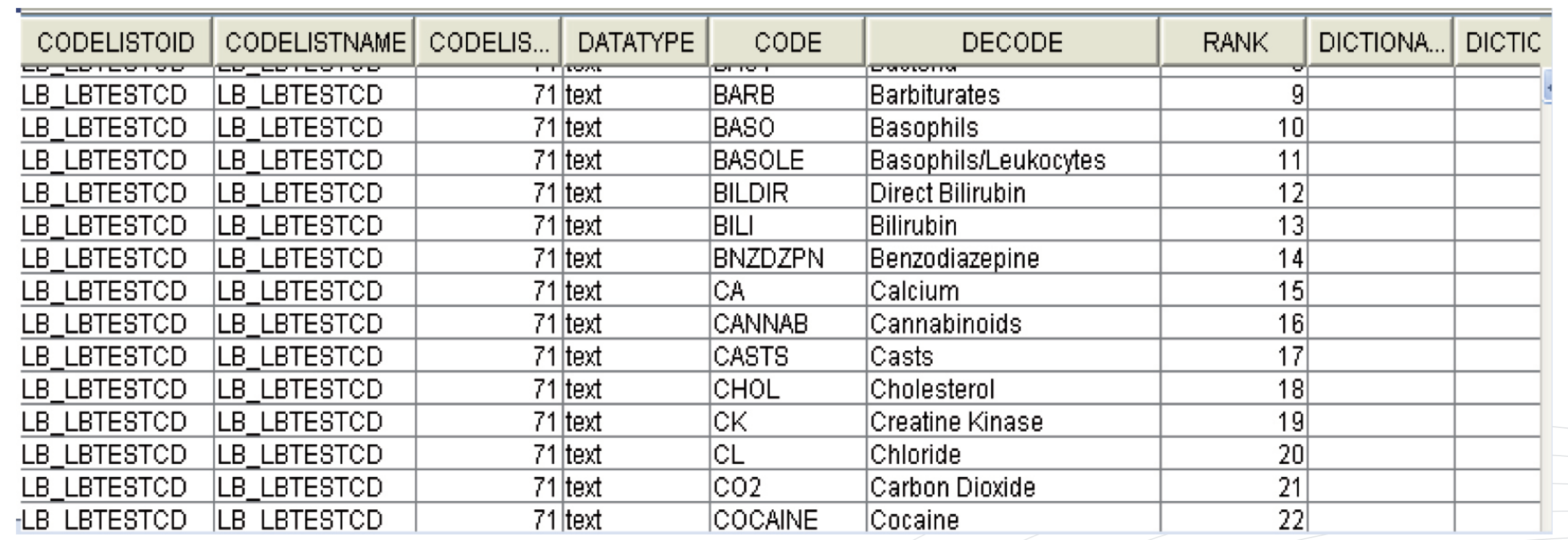
Code List Information Lists Possible Values: Variable Level Codes | Value Level Codes | Outside References

VARIABLE LEVEL DEFINE DATA



DATASET	VARIABLE	LABEL	DATAT...	L...	ORIGIN	ROLE	MAND...	COMMENT	CODELIS...	CODELIS...	DISPL
LB	LBORNRHI	Reference ...	text	40	EDT	VARIABLE ...	No				
LB	LBSTRESC	Character ...	text	200	Assigned	RESULT Q...	No				
LB	LBSTRESN	Numeric R...	float	6	Derived	RESULT Q...	No	Converted t...			6.3
LB	LBSTRESU	Standard U...	text	20	Assigned	VARIABLE ...	No	Associated...	LB_LBSTR...	LB_LBSTR...	
LB	LBSTNRLO	Reference ...	float	6	Assigned	VARIABLE ...	No				6.3
LB	LBSTNRHI	Reference ...	float	6	Assigned	VARIABLE ...	No				6.3
LB	LBSTNRC	Reference ...	text	40	EDT	VARIABLE ...	No				
LB	LBNRIND	Reference ...	text	40	EDT	VARIABLE ...	No		LB_LBNRI...	LB_LBNRI...	
LB	LBNAM	Vendor Na...	text	60	Assigned	RECORD ...	No				
LB	LBSPEC	Specimen ...	text	40	Assigned	RECORD ...	No		LB_LBSPEC	LB_LBSPEC	
LB	LBBLFL	Baseline Fl...	text	1	Derived	RECORD ...	No	Last readin...	YES	YES	
LB	VISITNUM	Visit Number	float	3	Derived	TIMING	No	Based on s...	LB_VISITN...	LB_VISITN...	3.1
LB	VISIT	Visit Name	text	40	Derived	TIMING	No	Based on s...	LB_VISIT	LB_VISIT	
LB	VISITDY	Planned St...	integer	2	Derived	TIMING	No	Based on s...			2.0
LB	LBDTC	Date/Time ...	text	19	EDT	TIMING	No		ISO 8601	ISO 8601	
LB	LBDY	Study Day ...	integer	2	Derived	TIMING	No	If visit = 1 th...			2.0
LB	LBTPT	Planned Ti...	text	40	Assigned	TIMING	No	Based on L...	LB_LBTPT	LB_LBTPT	
LB	LBTPTNUM	Planned Ti...	float	6	EDT	TIMING	No				6.3

CODE LIST LEVEL DEFINE DATA



CODELISTOID	CODELISTNAME	CODELIS...	DATATYPE	CODE	DECODE	RANK	DICTIONA...	DICTIC
LB_LBTESTCD	LB_LBTESTCD	71	text	BARB	Barbiturates	9		
LB_LBTESTCD	LB_LBTESTCD	71	text	BASO	Basophils	10		
LB_LBTESTCD	LB_LBTESTCD	71	text	BASOLE	Basophils/Leukocytes	11		
LB_LBTESTCD	LB_LBTESTCD	71	text	BILDIR	Direct Bilirubin	12		
LB_LBTESTCD	LB_LBTESTCD	71	text	BILI	Bilirubin	13		
LB_LBTESTCD	LB_LBTESTCD	71	text	BNZDZPN	Benzodiazepine	14		
LB_LBTESTCD	LB_LBTESTCD	71	text	CA	Calcium	15		
LB_LBTESTCD	LB_LBTESTCD	71	text	CANNAB	Cannabinoids	16		
LB_LBTESTCD	LB_LBTESTCD	71	text	CASTS	Casts	17		
LB_LBTESTCD	LB_LBTESTCD	71	text	CHOL	Cholesterol	18		
LB_LBTESTCD	LB_LBTESTCD	71	text	CK	Creatine Kinase	19		
LB_LBTESTCD	LB_LBTESTCD	71	text	CL	Chloride	20		
LB_LBTESTCD	LB_LBTESTCD	71	text	CO2	Carbon Dioxide	21		
LB_LBTESTCD	LB_LBTESTCD	71	text	COCAINE	Cocaine	22		

DEFINE DATA TO DEFINE.XML WITH PROC TEMPLATE

Moving from the six define data sets to the define.xml can be usefully broken into four parts facilitated by the use of PROC TEMPLATE: 1) Read in tagset names and data values , 2) Format and print tagset data, 3) Trigger the tagset events, and 4) Format and output printed tagset datasets as define.xml

READ IN TAGSET NAMES AND DATA VALUES

```
proc template;  define tagset allvars;  default_event="all";  Indent=3;
define event leaf;  set $tablename value;  end;
define event data;  set $data_values[name] value;  end;
```

The leaf event reads the tagset dataset during the trigger portion. The data event reads the variables per tagset event.

FORMAT AND PRINT TAGSET (def:ValueListDef tag)

```
define event vallevel;
     put "<def:ValueListDef OID=""" $data_values["DATASET"] "."
$data_values["VARIABLE"] """>" nl /if
     cmp('1',$data_values["ORDER"]);  **uppercase here;
     put "<ItemRef ItemOID=""" $data_values["DATASET"] "."
$data_values["VARIABLE"] "."   $data_values["VALUENAME"] """ OrderNumber="""
$data_values["VALUEORDER"] """" nl;
     put "Mandatory=""" $data_values["MANDATORY"] """
RoleCodeListOID=""RoleCodeList""></ItemRef>" nl;
     put "</def:ValueListDef>" nl /if cmp('YES',$data_values["END"]);
end;
```

The tagset format is precisely what is required by the xsl style sheet. The beginning and ending of the main tagset have a ‘cmp’ portion at the end. to compute the beginning and end of the main ValueListDef tagset. “ORDER” and “END” are ordering variables that allow nesting to occur.

TAGSET PRINT EXAMPLE

```
<def:ValueListDef OID="DA.DATESTCD">
<ItemRef ItemOID="DA.DATESTCD.DADISNO" OrderNumber="1"
Mandatory="No" RoleCodeListOID="RoleCodeList"></ItemRef>
<ItemRef ItemOID="DA.DATESTCD.DARETNO" OrderNumber="2"
Mandatory="No" RoleCodeListOID="RoleCodeList"></ItemRef>
</def:ValueListDef>
```

The attribute fields are populated by the define datasets. These nested tagsets then are processed by the extensible stylesheet to present the metadata in a readable form.

TRIGGER TAGSET EVENTS

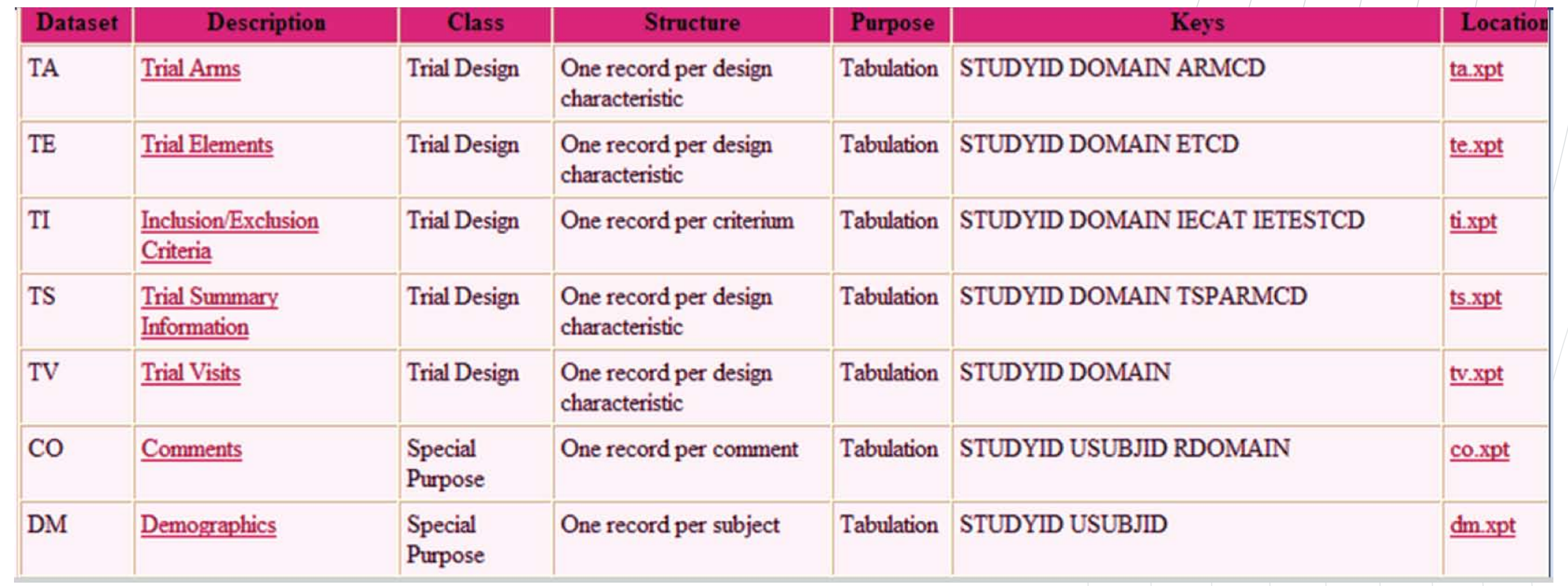
```
define event row;  start:
  unset $data_values; break; finish:
  break / if section ne "body";
  do / if $tablename eq "Data Set WORK.STUDY"; trigger study; done;    ...
  do / if $tablename eq "Data Set WORK.BOTTOM"; trigger bottom; done; end;
 output_type="xml";  nobreakspace=" ";mapsub=%nrstr("/&lt;;/&gt;/&amp;/");
 map=%nrstr("<>&"); end;
```

OUTPUT PRINTED TAGSET DATASETS AS DEFINE.XML

```
%macro printout(dset, variables, byvar);
data &dset.; retain &variables;  set &dset.; run;
proc print noobs data=&dset.;
  var  &variables;  by  &byvar notsorted; run;
%mend printout;
ods markup tagset=allvars file="&outfold./define.xml";
 %printout(study, FILEOID ORIGINATOR SOURCESYSTEM SOURCESYSTEMVERSION
STUDYOID STUDYNAME STUDYDESCRIPTION PROTOCOLNAME METADATAVERSION
METADATANAME METADATADES DEFINEVERSION STANDARDNAME
STANDARDVERSION);  /… /
%printout(bottom, FILEOID);
```

Another set of macro calls shows the order and content within each tagset. For the define.xml to work properly, it should be in a folder that contains all the documents, datasets it externally links, any external graphics used for visual functionality, and the proper extensible xsl stylesheet and css cascading stylesheet. It is trivial to adjust the colors and add a logo to the define.xml.

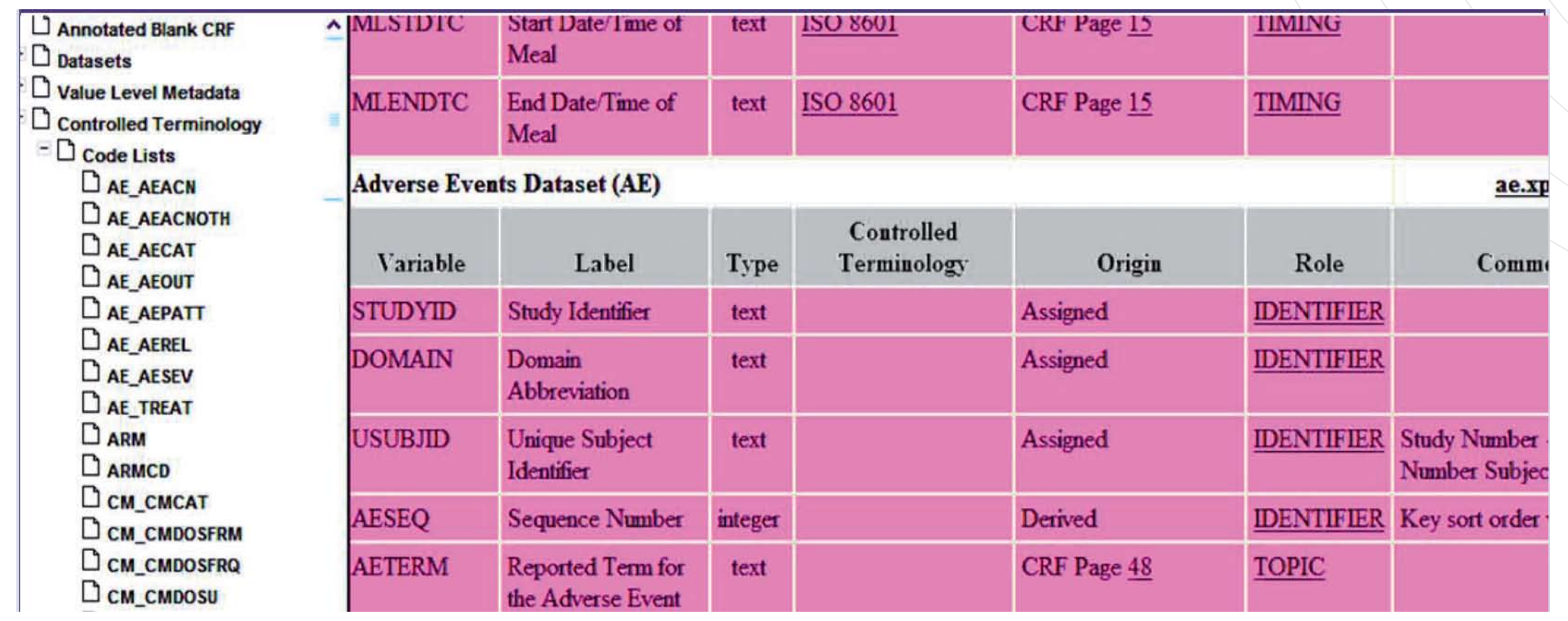
DEFINE.XML –DOMAIN LEVEL



Dataset	Description	Class	Structure	Purpose	Keys	Location
TA	Trial Arms	Trial Design	One record per design characteristic	Tabulation	STUDYID DOMAIN ARMCD	ta.xpt
TE	Trial Elements	Trial Design	One record per design characteristic	Tabulation	STUDYID DOMAIN ETCD	te.xpt
TI	Inclusion/Exclusion Criteria	Trial Design	One record per criterium	Tabulation	STUDYID DOMAIN IECAT IETESTCD	ti.xpt
TS	Trial Summary Information	Trial Design	One record per design characteristic	Tabulation	STUDYID DOMAIN TSPARMCD	ts.xpt
TV	Trial Visits	Trial Design	One record per design characteristic	Tabulation	STUDYID DOMAIN	tv.xpt
CO	Comments	Special Purpose	One record per comment	Tabulation	STUDYID USUBJID RDOMAIN	co.xpt
DM	Demographics	Special Purpose	One record per subject	Tabulation	STUDYID USUBJID	dm.xpt

There are linked levels of information in the define that correspond to each of the input define data sets. Not all information available in the define.xml is presented using the stylesheet, but all information can be viewed in native format by opening the file in WordPad or a similar application.

DEFINE.XML –VARIABLE LEVEL



MLSTDTC	Start Date/Time of Meal	text	ISO 8601	CRF Page 15	TIMING	
MLENDTC	End Date/Time of Meal	text	ISO 8601	CRF Page 15	TIMING	

Adverse Events Dataset (AE) ae.xp

Variable	Label	Type	Controlled Terminology	Origin	Role	Comm
STUDYID	Study Identifier	text		Assigned	IDENTIFIER	
DOMAIN	Domain Abbreviation	text		Assigned	IDENTIFIER	
USUBJID	Unique Subject Identifier	text		Assigned	IDENTIFIER	Study Number Number Subjec
AESEQ	Sequence Number	integer		Derived	IDENTIFIER	Key sort order
AETERM	Reported Term for the Adverse Event	text		CRF Page 48	TOPIC	

ACKNOWLEDGEMENTS

We want to thank Celerion for the support received for this work, and we want to express our gratitude to the CDISC community for the wealth of public information on the application of data standards.

www.celerion.com