CDISC Define-XML Specification Version 2.0

Prepared by

CDISC Define-XML Team

Notes to Readers

- This is the specification for Version 2.0 of the CDISC Define-XML standard.
- This is an update of the Case Report Tabulation Data Definition Version 1.0.0 Specification.

Revision History

Date	Version	Summary of Changes
2005-02-05	1.0.0	This is the official implementation version of the Case Report Tabulation Data Definition specification.
2005-02-09	1.0.0	Administrative update.
2009-11-27	2.0.0.1	Draft specification for Define-XML 2.0.0
2011-09-16	2.0.0.2	Updated Define-XML 2.0.0 specification in new format
2012-07-23	2.0.0.3	Update for SRC review and draft publication.
2012-08-31	2.0DRAFT	Public review DRAFT of version 2.0 of the Define-XML Standard.
2013-03-05	2.0.0	Production version of the Define-XML v2.0 Specification

Table of Contents

1. Introduction

1.1. Purpose of this document

This specification describes an updated Define-XML 2.0.0 model that is used to describe CDISC SDTM, SEND and ADaM datasets for the purpose of submissions to the FDA, as well as any proprietary (non-CDISC) dataset structure. Define-XML version 2.0.0 can be used to transmit metadata for the following CDISC standards:

- SDTM Implementation Guide Versions 3.1.2 and higher
- ADaM Implementation Guide Versions 1.0 and higher
- SEND Implementation Guide Versions 3.0 and higher

The Define-XML model is implemented using extensions to the CDISC Operational Data Model (ODM) XML schema. The intent is to fully comply with all applicable regulations and guidance. This document includes substantially all of the material in the initial specification for Define-XML as well as updates based on:

- Feedback from early adopters
- Changes to take advantage of ODM version 1.3.2
- Changes and clarifications required for compatibility with:
    - Version 3.1.2 of the CDISC Study Data Tabulation Model Implementation Guide (SDTM-IG)
    - Version 2.1 of the CDISC Analysis Dataset Model (ADaM)
    - Version 1.0 of the CDISC Analysis Dataset Model (ADaM) Implementation Guide (ADaM-IG)
- Recommendations from the CDISC SDTM Metadata team

Version 1.0.0 of Define-XML described the requirements for constructing *define.xml* documents to replace the *define.pdf* documents recommended in the FDA's 1999 *Regulatory Submissions in Electronic Format, General Considerations* guidance. Since that time, Define-XML documents have proven to be a useful mechanism for transmission of case report tabulation (CRT) metadata. One of the key benefits to FDA reviewers is that this format provides both a machine readable format for use by the various FDA software applications and, through the provision of an XSL stylesheet, a browser-based report describing the contents of a clinical study.

1.2. U.S. Electronic Submission Background

In the United States, the approval process for regulated human and animal health products requires the submission of data from clinical trials and other studies as expressed in the Code of Federal Regulations (CFR). The FDA established the regulatory basis for wholly electronic submission of data in 1997 with the publication of regulations on the use of electronic records in place of paper records (21 CFR Part 11). In 1999, the FDA standardized the submission of clinical and non-clinical data using the SAS Version 5 XPORT Transport Format and the submission of metadata using Portable Document Format (PDF), respectively. In 2005, the *Study Data Specifications* published by the FDA included the recommendation that data definitions (metadata) be provided as a Define-XML file. In December 2011, the *CDER Common Data Standards Issues Document* stated that "a properly functioning *define.xml* file is an important part of the submission of standardized electronic datasets and should not be considered optional."

1.3. CDISC

The Clinical Data Interchange Standards Consortium (CDISC) is a non-profit organization whose mission is to develop and support global, platform-independent data standards that enable information system interoperability to improve medical research and related areas of healthcare.
The FDA has collaborated with CDISC since its founding in order to standardize the content and structure of clinical trials and non-clinical study data for regulatory submission. CDISC sponsors and members represent more than 250 companies active in the research and development of regulated health-related products.

1.4. Operational Data Model (ODM)

The Define-XML standard is based on the CDISC Operational Data Model (ODM) XML schema. ODM is a vendor neutral, platform independent format for the interchange and archival of clinical study data. The model includes the clinical data along with its associated metadata, administrative data, reference data and audit information. All of the information that needs to be shared among different software systems during the setup, operation, analysis, submission or for long term retention as part of an archive is included in the model. ODM has been embraced by a broad range of clinical development organizations, and a number of vendors provide software applications and tools that use ODM. The current version of the ODM standard is available at http://www.cdisc.org/odm.

One of the features of the ODM is a standardized mechanism for defining schema extensions to provide functionality needed to support interchange requirements for specialized use cases. To address the specific needs of data transmission in support of regulatory submissions, CDISC has developed the Define-XML model, which is implemented as a set of extensions to the base ODM schema. These extensions follow the guidelines for Vendor Extensions provided in the ODM specification and comply with the W3C XML Schema 1.0 specification. The XML schema files for the Define-XML standard are available online at http://www.cdisc.org/define-xml.

While this document is intended to be understandable to readers with minimal technical knowledge of the ODM and XML, knowledge of this document alone is not a substitute for knowledge of the ODM nor is it sufficient to produce complete Define-XML files. This document should be used in close concert with the current version of the ODM specification as well as current versions of the relevant data standards. The ODM specification package is available online at http://www.cdisc.org/odm. Numerous examples of XML fragments appear in this document. Many of these examples are provided as XML files and can be downloaded from the CDISC website (http://www.cdisc.org/define-xml).

1.5. Study Data Tabulation Model (SDTM)

The CDISC Study Data Tabulation Model (SDTM) defines a standard structure for case report form data tabulations that are required to be submitted as part of a product application to the FDA. The CDISC SDTM is used to submit clinical trial and non-clinical study data for product applications across all therapeutic areas. For a current list of data and metadata requirements for FDA submission see http://www.fda.gov/Drugs/DevelopmentApprovalProcess/FormsSubmissionRequirements/ElectronicSubmissions. The current version of the SDTM standard is available at http://www.cdisc.org/sdtm. Define-XML version 2.0.0 can be used to transmit metadata for the SDTM Implementation Guide Versions 3.1.2 and higher.

1.6. Analysis Data Model (ADaM)

The CDISC Analysis Data Model (ADaM) defines standards for analysis datasets that are submitted as part of a product application to the FDA. In addition to defining fundamental principles that apply to all analysis datasets, ADaM defines standard structures that are appropriate for the majority of analysis datasets. Because analysis datasets are developed to support specific analyses, ADaM has additional metadata that is not found in SDTM or SEND, notably analysis results metadata. Additionally, the metadata to describe variable sources and derivations is of primary importance. The current version of the ADaM standard is available at http://www.cdisc.org/adam. Define-XML version 2.0.0 can be used to transmit metadata for the ADaM Implementation Guide Versions 1.0 and higher.

2. Abbreviations and References

2.1. Definitions and Abbreviations

ADaM	Analysis Dataset Model - developed by CDISC.
ADaM-IG	Analysis Dataset Model Implementation Guide – developed by CDISC
CRF	Case Report Forms
CRT	Case Report Tabulations
eCTD	Electronic Common Technical Document
FDA	United States Food and Drug Agency
ICH	International Conference on Harmonization of technical requirements for registration of pharmaceuticals for human use.
ODM	Operational Data Model – developed by CDISC as an XML format for the transmission and archival of clinical trials data and metadata.
OID	ODM element identifier.
PDF	Portable Document Format – an open standard for document exchange developed by Adobe Systems
SAP	Statistical Analysis Plan
SDTM	Study Data Tabulation Model - developed by CDISC for the purpose of submitting Case Report Form tabulations to the United States Food and Drug Agency
SDTM-MSG	Study Data Tabulation Model Metadata Submission Guidelines – developed by CDISC
SDTM-IG	Study Data Tabulation Model Implementation Guide – developed by CDISC
SEND	Standard for Exchange of Non-clinical Data – developed by CDISC

URI	Uniform Resource Identifier - a string of characters used to identify a resource on the internet
URL	Uniform Resource Locator
W3C	World Wide Web Consortium
XLink	XML Linking Language – developed by the W3C
XML	Extensible Markup Language - developed by the W3C
XPT	SAS Transport file – an open standard for data transmission developed and maintained by SAS
XSL	Extensible Stylesheet Language – developed by the W3C for the purpose of transforming and formatting XML documents

2.2. **References**

The documents referenced during the development of this Define-XML Specification may be accessed via the links provided below.

- CDISC website
  http://www.cdisc.org
- ODM Version 1.3.2
  http://www.cdisc.org/odm
- CRT-DDS - Case Report Tabulation Data Definition Specification (define.xml) Version 1.0
  http://www.cdisc.org/models/def/v1.0/index.html
- SDTM - Study Data Tabulation Model (SDTM) Final Version 1.2
  http://www.cdisc.org/sdtm
- SDTM-IG - CDISC SDTM Implementation Guide Version 3.1.2
  http://www.cdisc.org/sdtm
- SDTM-MSG – SDTM Metadata Submission Guidelines V1
  http://www.cdisc.org/sdtm
- ADAM - CDISC ADaM Analysis Data Model Version 2.1
  http://www.cdisc.org/adam
- ADAM-IG - CDISC ADaM Implementation Guide Version 1.0
  http://www.cdisc.org/adam
- SEND-IG - CDISC SEND Implementation Guide Version 3.0
  http://www.cdisc.org/send
- Controlled Terminology
  http://www.cancer.gov/cancertopics/cancerlibrary/terminologyresources/cdisc
- XML Schema Validation for Define.xml White Paper
  http://www.cdisc.org/define-xml
- FDA eCTD Guidance - Electronic Common Technical Document
  http://www.fda.gov/downloads/Drugs/GuidanceComplianceRegulatoryInformation/Guidances/UCM072349.pdf
- FDA Study Data Specifications
  http://www.fda.gov/downloads/ForIndustry/DataStandards/StudyDataStandards/UCM312964.pdf
- FDA Study Data Standards Page
  http://www.fda.gov/forindustry/datastandards/studydatastandards/default.htm

3. Conformity and General Issues

This section supplements the corresponding section, "General Issues", of the ODM v1.3.2 specification. All conformity requirements described in the ODM v1.3.2 specification are also applicable to Define-XML files as they are based on the ODM 1.3.2 model.

3.1. File Conformity

The namespace URI for version 2.0.0 of Define-XML is:
http://www.cdisc.org/ns/def/v2.0

Throughout this document, the following conventions are used for namespaces:

- ODM elements and attributes are in the default namespace (i.e. they have no namespace prefix)
- Define-XML elements use the namespace prefix "def"
- Define-XML attributes use the namespace prefix "def" only if they appear within ODM elements

Note that these namespace prefixes are used throughout this document and are recommended as best practice to make it easier for users to understand and implement Define-XML, and aid in the comparison of documents. In practice other namespace prefixes can be used as long as the *define.xml* conforms to the rules of XML Namespaces.

Any XML included in a Define-XML document that is not described in this specification is considered an extension.

Deprecated elements or attributes are not valid for use and are considered errors.

3.2. Extensions

The Define-XML schema permits vendor extensions, as defined in the ODM 1.3.2 specification, to the elements defined in this specification. These extensions may take the form of CDISC-created extensions, such as the Analysis Results Metadata extension or vendor extensions. Any XML not explicitly specified as part of Define-XML v2.0.0 is considered an extension. This includes ODM metadata not explicitly referenced in this specification. Extensions have no implied meaning with respect to the Define-XML standard; the sender and receiver must agree on a meaning themselves. That is, *define.xml* files that use extensions are not wrong, but instead the extensions may be ignored unless the sender and receiver have agreed otherwise. This also means that validators should flag ODM metadata not explicitly mentioned in the Define-XML specification with informational messages, and not with errors or warnings.

Requirements for vendor extensions to the Define-XML schema are:

- The vendor must supply an XML Schema fully describing their extended Define-XML format if it uses extended elements or attributes not already defined in the ODM namespace or Define-XML extension.

- Extended Define-XML files should reference the proper extension Schema.

- The extension may add new XML elements and attributes, but may not render any standard Define-XML elements or attributes obsolete. Vendor extensions may not be used for information that is normally expressed using other Define-XML elements or attributes.

- Elements and attributes from the ODM schema that are not a part of the Define-XML schema can be used as extensions, but no elements or attributes can be added to the ODM namespace.

- All extension elements and attributes not already defined in the ODM namespace must use a distinct XML namespace to ensure that there are no naming conflicts with other vendor extensions.

- The meaning of a Define-XML file must not be fundamentally changed by the addition of extensions.

- Removing all vendor extensions from an extended *define.xml* file must result in a valid, meaningful and accurate *define.xml* file.

- Vendors should be able to produce *define.xml* files free of any vendor extensions upon request.

3.3. Define-XML Document Structure

The example below shows the XML that would comprise the minimal structure of any ODM 1.3.2 document that contains a Define-XML document. It illustrates a valid Define-XML document header and the gray box illustrates the set of elements that comprise this standard in the order in which they should appear in a valid Define-XML file.

Example 3.3: Define-XML document structure.

```
<?xml version="1.0" encoding="UTF-8"?>
<ODM xmlns="http://www.cdisc.org/ns/odm/v1.3"
  xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:def="http://www.cdisc.org/ns/def/v2.0"
  FileType="Snapshot" ODMVersion="1.3.2"
  FileOID="Studycdisc01-Define-XML_2.0.0"
  CreationDateTime="2012-06-21T11:07:23-05:00"
  Originator="CDISC XML Technologies Team">
  <Study OID="cdisc01">
    <GlobalVariables>
      <StudyName>CDISC01</StudyName>
      <StudyDescription>CDISC Test Study</StudyDescription>
      <ProtocolName>CDISC 01</ProtocolName>
    </GlobalVariables>
    <MetaDataVersion OID="MDV.CDISC01.SDTMIG.3.1.2.SDTM.1.2"
      Name="Study CDISC01, Data Definitions"
      Description="Study CDISC01, Data Definitions"
      def:DefineVersion="2.0.0"
      def:StandardName="CDISC SDTM"
      def:StandardVersion="3.1.2">
        < Annotated Case Report Forms (def:AnnotatedCRF) >
        < Supplemental Data Definitions (def:SupplementalDoc) >
        < Value Level Metadata (def:ValueListDef) >
        < Where Clause Definitions (def:WhereClauseDef) >
        < Domain Level Metadata (ItemGroupDef) >
        < Variable Level Metadata (ItemDef) >
        < Controlled Terminology Metadata (CodeList) >
        < Computational Algorithms (MethodDef) >
        < Comments (def:CommentDef) >
        < Referenced Documents (def:leaf) >
    </MetaDataVersion>
  </Study>
</ODM>
```

3.4. Ordering Elements and Attributes

3.4.1. Use of the OrderNumber Attribute

In several locations, the Define-XML specification uses an optional *OrderNumber* attribute to define the relative ordering of elements within a container. In the context of Define-XML, if the *OrderNumber* attribute is specified on an element, all elements of the same type (element name) within the same parent element must also specify an *OrderNumber* attribute. It is an error to mix elements with *OrderNumber* and without *OrderNumber* within a given parent element. If no *OrderNumber* attribute is specified in cases where order is important, the ordering is derived from the XML document order.

3.4.2. Other Order Considerations for Elements

The Define-XML specification does not require that *ItemGroupDef* elements related to the SDTM follow a particular order. However, for purposes of regulatory submissions, *ItemGroupDef* elements should follow the order recommended in the SDTM-MSG. Note that the SDTM-MSG indicates that the datasets should be displayed in the following Class order:

- TRIAL DESIGN
- SPECIAL PURPOSE - Subject-Level
- INTERVENTIONS
- EVENTS
- FINDINGS
- FINDINGS ABOUT – not yet added to the NCI/CDISC Controlled Terminology as of the time of writing this document
- RELATIONSHIP

For regulatory submissions of ADaM datasets, a standard order of display has not been established. However, the following Class order seems reasonable, as it is consistent with the ordering of Class values in ADaM 2.1:

- SUBJECT LEVEL ANALYSIS DATASET
- ADVERSE EVENTS ANALYSIS DATASET – not yet added to the NCI/CDISC Controlled Terminology as of the time of writing this document
- BASIC DATA STRUCTURE - datasets in alphabetical order by dataset name
- ADAM OTHER

As additional ADaM dataset classes are defined, it would be reasonable to insert them between Basic Data Structure and ADaM Other.

Within each class the datasets should be displayed in ascending alphabetic order by Name as maintained in the value of the ItemGroupDef Name attribute.

3.5. Defs and Refs

This document sometimes references elements as *Def*s and *Ref*s. In Define-XML, an element whose name ends with "Def" is the declaration of an object instance. An element whose name ends with "Ref" is a reference to that object from some other entity.

For example, in this specification variables are declared using elements named *ItemDef*. However, to indicate that an *ItemGroup* includes a particular item instance, an *ItemRef* element is used to reference the appropriate *ItemDef* object.

3.5.1. OIDs

Attributes whose names end with "OID", (or sometimes called "OIDs"), are used to uniquely identify the different metadata objects. The Define-XML specification does not mandate a format for OIDs. OIDs are only intended as a mechanism for unambiguously linking between a definition of an object and references to it. The examples in this document use prefixes at the start of OIDs to indicate the object type, however this is not required. It is equally valid to use randomly generated identifiers.
Example 4.4.2.4 illustrates the concepts of linking between a definition of an object and references to it.

OIDs play a critical role in presenting the *define.xml* file as one interconnected document containing a web of multiple parts that are linked together to form one set of metadata. OIDs provide these links. They allow us to state the variables of a dataset, and yet define those variables outside of the dataset context. They allow us to name a list of allowable values for a

variable, and yet define those lists of values outside of the variable context. Because definitions can exist outside of any context, any one definition can be referenced from multiple contexts. For example, the variable STUDYID is found in almost every SDTM data set. By defining STUDYID outside of any dataset context, we can define it only once and reference the definition from every dataset that contains the variable. In this sense we can say that the STUDYID definition is being *shared* or *reused* by all the datasets.

The value of an OID attribute has no intrinsic meaning by itself (although some may find convenience in applying a pre-defined convention for assigning values, as is illustrated throughout this document), but its significance comes in its matching with values of other OIDs. For example, the definition of a variable referenced in an *ItemRef* element contained within an *ItemGroupDef* element can be found in the one and only *ItemDef* element (outside of *ItemGroupDef*) whose OID attribute value matches the value of the ItemRef *ItemOID* attribute. Similar associations are made between variables and their corresponding codelist definitions, variables and their corresponding valuelist definitions, and in other sections of the document. As you study the examples throughout this document, take note of these OID value pairings.

3.6. **Validation of a Define-XML v2.0.0 document**

A valid *define.xml* file must

- Properly reference versions of the CDISC standards.
- Be well formed and conform to the XML schemas.
- Meet all of the requirements documented in this specification.

Once a *define.xml* file is valid according to the schema, validation software should consider all other Define-XML requirements in the specification. These include rules about conditionally required components or other business rules in this document. The Define-XML schema can only enforce some of the standard, so this additional level of validation is required to determine if a *define.xml* document is compliant with Define-XML v2.0.0. See the XML Schema Validation for Define.xml White Paper for additional information.

The correct ordering of elements within a document is an absolute requirement for the document to be valid with respect to the Define-XML schema. The use of an XML-Schema definition and a validating parser environment makes detection of improperly ordered content fairly straightforward. In the absence of such mechanisms, care should be extended to following the order specified by the documentation for all extension content.

Note that XML is case sensitive, and case sensitivity plays a role in creating a valid *define.xml* file. For example, def:Class="findings" is not valid, but def:Class="FINDINGS" is valid because it uses the capitalized version of "FINDINGS" included in this specification.

4. General Specifications for Define-XML

The purpose of Define-XML is to support the interchange of dataset metadata for clinical research applications in a machine-readable format. An important use case for Define-XML is to support the submission of clinical trials data in CDISC SDTM, SEND or ADaM format to regulatory authorities. The key metadata components to support submissions are:

- Dataset definitions
- Dataset variable definitions
- Controlled Terminology definitions
- Value list definitions
- Links to supporting documents
- Computational method definitions
- Comments definitions

4.1. Dataset Definitions

CDISC SDTM, SEND and ADaM study datasets are modeled as tables where the columns represent variables and the rows represent observed or derived values of those variables. Dataset metadata is represented in Define-XML as an *ItemGroupDef* element. A detailed specification is provided in section 5.3.10.

Dataset definitions include references to the definitions of dataset variables and, when used for regulatory submissions, a reference to the file containing the dataset data content. In the Define-XML, the references to dataset variables are specified using the *ItemRef* element (section 5.3.8.1) and the reference to the dataset file is provided by a *def:leaf* element (section 5.3.15). Sponsors may provide SDTM comments or ADaM Documentation references at the dataset level. Comments are provided in the Define-XML using the *def:CommentDef* element (section 5.3.14).

Currently SDTM, SEND and ADaM study datasets are submitted to regulatory authorities as SAS Transport files. Although a dataset domain normally corresponds to a single SAS Transport file, in practice for SDTM datasets there are cases where dataset domains may be split into two or more files. When this occurs, each dataset file must be represented by a separate *ItemGroupDef* in the Define-XML. The *Name* attribute in each of the *ItemGroupDef* elements must contain the name of the split dataset and *SASDatasetName* attribute must match the name of the specific SAS Transport file (without the .xpt extension). The *Domain* attribute must be provided and must match the value of the DOMAIN variable in the corresponding dataset. An example of a Define-XML representation of a split dataset is provided in Section 4.1.1.3. Further requirements for splitting SDTM domains are provided in the SDTM-IG.

4.1.1. Examples of Dataset Metadata in Define-XML

Note that most examples in this document are based on the SDTM-MSG V1 published example. These examples were originally based on Define-XML v1.0.0 and they have been updated to reflect the Define-XML v2.0.0 standard.

4.1.1.1.Example of Non-Repeating dataset / Demographics Domain Dataset

This is an example of an *ItemGroupDef* element that represents the SDTM Demographics (DM) domain dataset metadata. The SDTM DM domain has only one record for each subject so the *Repeating* attribute is set to No. This ItemGroupDef contains one *ItemRef* element for each variable in the Demographics Domain. For variables that are designated as Required in the SDTM-IG, the *Mandatory* attribute should be set to Yes. For variables that are designated by SDTM-IG as Expected or Permissible, the *Mandatory* attribute should normally be set to No. If, however, the sponsor chooses to designate as Required a variable that the SDTM-IG designates as Permissible or Expected, then the *Mandatory* attribute should be set to Yes. In the example below, AGE (designated as Expected by SDTM-IG) and ETHNIC (designated as Permissible by SDTM-IG) are both set to Mandatory due to the sponsor's decision to designate both as Required. The *Domain* attribute of the *ItemGroupDef* element is mandatory in the context of a regulatory submission and has been set to the Domain as specified by the SDTM Metadata Submission Guidelines.

```
<!-- Dataset Definition (DM) -->
<ItemGroupDef OID="IG.DM
  Domain="DM"
  Name="DM"
  Repeating="No"
  IsReferenceData="No"
  SASDatasetName="DM"
  Purpose="Tabulation"
  def:Structure="One record per subject"
  def:Class="SPECIAL PURPOSE"
  def:CommentOID="COM.DOMAIN.DM"
  def:ArchiveLocationID="LF.DM">
  <Description>
    <TranslatedText xml:lang="en">Demographics</TranslatedText>
  </Description>
  <ItemRef ItemOID="IT.STUDYID" OrderNumber="1" Mandatory="Yes" KeySequence="1"/>
  <ItemRef ItemOID="IT.DM.DOMAIN" OrderNumber="2" Mandatory="Yes"/>
  <ItemRef ItemOID="IT.USUBJID" OrderNumber="3" Mandatory="Yes" KeySequence="2"
    MethodOID="MT.USUBJID"/>
  <ItemRef ItemOID="IT.DM.SUBJID" OrderNumber="4" Mandatory="Yes"/>
  <ItemRef ItemOID="IT.DM.RFSTDTC" OrderNumber="5" Mandatory="No"
    MethodOID="MT.RFSTDTC"/>
  <ItemRef ItemOID="IT.DM.RFENDTC" OrderNumber="6" Mandatory="No"
    MethodOID="MT.RFENDTC"/>
  <ItemRef ItemOID="IT.DM.SITEID" OrderNumber="7" Mandatory="Yes"/>
  <ItemRef ItemOID="IT.DM.BRTHDTC" OrderNumber="8" Mandatory="No"/>
  <ItemRef ItemOID="IT.DM.AGE" OrderNumber="9" Mandatory="Yes"
    MethodOID="MT.AGE"/>
  <ItemRef ItemOID="IT.DM.AGEU" OrderNumber="10" Mandatory="No"/>
  <ItemRef ItemOID="IT.DM.SEX" OrderNumber="11" Mandatory="Yes"/>
  <ItemRef ItemOID="IT.DM.RACE" OrderNumber="12" Mandatory="No"/>
  <ItemRef ItemOID="IT.DM.ETHNIC" OrderNumber="13" Mandatory="Yes"/>
  <ItemRef ItemOID="IT.DM.ARMCD" OrderNumber="14" Mandatory="Yes"/>
  <ItemRef ItemOID="IT.DM.ARM" OrderNumber="15" Mandatory="Yes"/>
  <ItemRef ItemOID="IT.DM.COUNTRY" OrderNumber="16" Mandatory="Yes"/>
  <def:leaf ID="LF.DM" xlink:href="dm.xpt">
  <def:title>dm.xpt</def:title>
  </def:leaf>
</ItemGroupDef>
```

4.1.1.2. Example of Reference Dataset / Trial Elements Domain Dataset

This example illustrates the use of the *ItemGroupDef* element to provide metadata for the SDTM Trial Elements (TE) domain dataset. The SDTM TE domain does not contain subject level data so the *IsReferenceData* attribute is included and is set to "Yes".

```
<!-- Dataset Definition (TE) -->
<ItemGroupDef OID="IG.TE"
  Domain="TE"
  Name="TE"
  Repeating="No"
  IsReferenceData="Yes"
  SASDatasetName="TE"
  Purpose="Tabulation"
  def:Structure="One record per planned Element"
  def:Class="TRIAL DESIGN"
  def:ArchiveLocationID="LF.TE">
  <Description>
    <TranslatedText xml:lang="en">Trial Elements</TranslatedText>
  </Description>
  <ItemRef ItemOID="IT.STUDYID" OrderNumber="1" Mandatory="Yes" KeySequence="1"/>
  <ItemRef ItemOID="IT.TE.DOMAIN" OrderNumber="2" Mandatory="Yes"/>
  <ItemRef ItemOID="IT.TE.ETCD" OrderNumber="3" Mandatory="Yes" KeySequence="2"/>
  <ItemRef ItemOID="IT.TE.ELEMENT" OrderNumber="4" Mandatory="Yes"/>
  <ItemRef ItemOID="IT.TE.TESTRL" OrderNumber="5" Mandatory="Yes"/>
  <ItemRef ItemOID="IT.TE.TEDUR" OrderNumber="6" Mandatory="No"/>
  <def:leaf ID="LF.TE" xlink:href="te.xpt">
  <def:title>te.xpt</def:title>
  </def:leaf>
</ItemGroupDef>
```

4.1.1.3. Example of a Split SDTM dataset definition

This example illustrates the construction of *ItemGroupDef* elements to provided metadata for two split datasets for the Questionnaire (QS) SDTM domain. The value of the *Domain* element for both *ItemGroupDef* elements is "QS". The *Name* and *SASDatasetName* attributes contain the split dataset names. Each *ItemGroupDef* element includes a *Description* child element with a label for the split dataset, and an *Alias* child element with the *Context* attribute set to "DomainDescription" to indicate that its *Name* attribute provides a label for the full domain dataset.

Each *ItemGroupDef* element includes attributes for the *def:ArchiveLocationID* and the *def:CommentOID*. These reference a *def:leaf* child element describing the location of the dataset file and a *def:Comment* element respectively.

The datasets follow the naming recommendations specified in Section 4.1.1.7 of the SDTM-IG. To save space in this document, not all *ItemRef* elements are included in this example.

Note that since split domain datasets may be combined by the data recipient, the variables belonging to each split must be defined in a consistent manner. *ItemRef* definitions may be reused, as illustrated by the use of the same *ItemOID* values across the two datasets, but if the variables differ in any of their metadata (e.g. Value Level metadata) across the split, separate *ItemRef* elements corresponding to distinct *ItemDef* elements are needed.

```
<!-- Dataset Definition (QSCG) -->
<ItemGroupDef OID="IG.QSCG"
  Domain="QS"
  Name="QSCG"
  Repeating="Yes"
  IsReferenceData="No"
  SASDatasetName="QSCG"
  Purpose="Tabulation"
  def:Structure="One record per questionnaire per question per visit per subject"
  def:Class="FINDINGS"
  def:CommentOID="COM.DOMAIN.QS"
  def:ArchiveLocationID="LF.QSCG">
  <Description>
    <TranslatedText xml:lang="en">Questionnaire-QSCG</TranslatedText>
  </Description>
  <ItemRef ItemOID="IT.STUDYID" OrderNumber="1" Mandatory="Yes" KeySequence="1"/>
  …
  <ItemRef ItemOID="IT.QS.QSDY" OrderNumber="17" Mandatory="No"
    MethodOID="MT.QSDY"/>
  <Alias Context="DomainDescription" Name="Questionnaires"/>
  <def:leaf ID="LF.QSCG" xlink:href="qscg.xpt">
  <def:title>qscg.xpt</def:title>
  </def:leaf>
</ItemGroupDef>
```

```
<!-- Dataset Definition (QSCS) -->
<ItemGroupDef OID="IG.QSCS"
  Domain="QS"
  Name="QSCS"
  Repeating="Yes"
  IsReferenceData="No"
  SASDatasetName="QSCS"
  Purpose="Tabulation"
  def:Structure="One record per questionnaire per question per visit per subject"
  def:Class="FINDINGS"
  def:CommentOID="COM.DOMAIN.QS"
  def:ArchiveLocationID="LF.QSCS">
  <Description>
    <TranslatedText xml:lang="en">Questionnaire-QSCS</TranslatedText>
  </Description>
  <ItemRef ItemOID="IT.STUDYID" OrderNumber="1" Mandatory="Yes" KeySequence="1"/>
  …
  <ItemRef ItemOID="IT.QS.QSDY" OrderNumber="17" Mandatory="No"
    MethodOID="MT.QSDY"/>
  <ItemRef ItemOID="IT.QS.QSEVLINT" OrderNumber="18" Mandatory="No"/>
  <Alias Context="DomainDescription" Name="Questionnaires"/>
  <def:leaf ID="LF.QSCS" xlink:href="qscs.xpt">
  <def:title>qscs.xpt</def:title>
  </def:leaf>
</ItemGroupDef>
<!-- Dataset Definition (QSMM) -->
<ItemGroupDef OID="IG.QSMM" Domain="QS"


…

<!-- Comment Definition: Long Comment, included in a PDF file -->
<def:CommentDef OID="COM.DOMAIN.QS">
  <Description>
    <TranslatedText xml:lang="en"> QS is submitted as a split dataset. The split
was done based on QSCAT as QSCG (CLINICAL GLOBAL IMPRESSIONS), QSCS (CORNELL SCALE
FOR DEPRESSION IN DEMENTIA) and QSMM (MINI MENTAL STATE EXAMINATION). See
additional documentation in the Reviewer's Guide, Split Datasets Section.
    </TranslatedText>
  </Description>
  <def:DocumentRef leafID="LF.ReviewersGuide"/>
</def:CommentDef>
```

4.2. **Dataset Variable Definitions**

The CDISC SDTM, SEND and ADaM specifications each define variable metadata requirements. Define-XML represents variable metadata using an *ItemDef* element. A detailed specification is provided in section 5.3.11. Variables are associated with datasets through ItemRefs contained in *ItemGroupDef* elements. Additional variable metadata is associated with ItemDefs and ItemRefs using the elements listed in the table below:

MetaData	Define-XML element	Reference
Controlled Terminology	CodeList	ItemDef/CodeListRef/@CodeListOID
Value Level Metadata	def:ValueListDef	ItemDef/def:ValueListRef/@ValueListOID
Computational Method	MethodDef	ItemRef/@MethodOID
Comments	def:CommentDef	ItemDef/@def:CommentOID
Origin	def:Origin	ItemDef/def:Origin

Note that Variables can have both a CodeList and a ValueList attached as they provide semantically distinct information. A CodeList provides a list of allowable values. A ValueList is used to define metadata based on the value of another variable (Value Level Metadata) to support data review and analysis in cases where variable metadata is not sufficient. See section 4.4 Value Level Metadata Definitions for further details.

4.2.1. Data Type Considerations

CDISC SDTM, SEND and ADaM variable data types are specified as either character or numeric ("Char" and "Num", respectively). Define-XML, because it is based on the CDISC ODM, supports a richer set of data types. The following table illustrates the mappings between Define-XML and SDTM data types:

Define-XML Data Type	Submission Data Type	Length	Considerations
text	Char	Maximum allowable length.	SAS Version 5 transport files restrict variable lengths to 200 characters.
integer	Num	The largest allowable integer width.	Use for numeric or equivalent variables that have discrete whole values (non-fractional). Can be positive, negative, or zero. ADaM date variables, are provided as integers.
float	Num	The largest allowable whole number width plus the maximum number of decimal digits.	Use for numeric variables that may contain a fractional component. It represents the set of all the decimal numbers with arbitrary lengths.
datetime	Char	N/A	Use if values for SDTM or Send variable represent Date Times (YYYY-MM-DDTHH:MM:SS.SS).
date	Char	N/A	Use if values for SDTM or SEND variable represent complete (YYYY-MM-DD) dates.

Define-XML Data Type	Submission Data Type	Length	Considerations
time	Char	N/A	Use if values for SDTM or SEND variable represent complete (HH:MM:SS.SS) times in ISO-8601 format.
partialDate	Char	N/A	Use when date values may be right truncated. That is, if the day or the day and month may be missing.
partialTime	Char	N/A	Use when time values may be right truncated. That is, if the seconds or seconds and minutes may be truncated.
partialDatetime	Char	N/A	Use when date or time values may be right truncated (see partialDate and partialTime). For example, use to represent a full date and time in HH:MM.
incompleteDatetime	Char	N/A	Use when date values may be missing a component but is not a partialdatetime.
durationDatetime	Char	N/A	Use to indicate values use the ISO8601 P notation to indicate a duration value.

Note: The date and time data types represent the planned specificity of the collected data, and not an interpretation of the actual collected values.

4.2.2. Examples of Variable Definitions

4.2.2.1. Example of SDTM Variables that are Domain Keys

This example illustrates how to reference a variable (Item) definition with a dataset (ItemGroup) definition using an *ItemRef* child element in an *ItemGroupDef* element. It also shows how to define the variable metadata using an *ItemDef* element.

Note that the OID value in the *ItemDef* elements matches the *ItemOID* values in the *ItemRef* elements.

Note that the Domain Keys are specified by the KeySequence attribute in the *ItemRef* element.

Note that the *Role* attribute is not included in the definition of the STUDYID variable since this attribute is optional and the Role is known for variables in SDTM standard domains.

```
<!-- Dataset Definition (DM) -->
<ItemGroupDef OID="IG.DM
  Domain="DM"
  Name="DM"
  Repeating="No"
  …
  def:ArchiveLocationID="LF.DM">
  <Description>
    <TranslatedText xml:lang="en">Demographics</TranslatedText>
  </Description>
  …
  <ItemRef ItemOID="IT.STUDYID" OrderNumber="1" Mandatory="Yes" KeySequence="1"/>
  <ItemRef ItemOID="IT.DM.DOMAIN" OrderNumber="2" Mandatory="Yes"/>
  <ItemRef ItemOID="IT.USUBJID" OrderNumber="3" Mandatory="Yes" KeySequence="2"
    MethodOID="MT.USUBJID"/>
  …
</ItemGroupDef>

  …

  <!-- Item Definition: Variable Level (STUDYID) -->
  <ItemDef OID="IT.STUDYID" Name="STUDYID" DataType="text" Length="7"
    SASFieldName="STUDYID">
    <Description>
      <TranslatedText xml:lang="en">Study Identifier</TranslatedText>
    </Description>
    <def:Origin Type="Protocol"/>
  </ItemDef>
```

4.2.2.2. Example of a Variable with Origin "CRF"

This example illustrates the child element *def:Origin* in the *ItemDef* element to provide a reference to a page in the Annotated CRF for a study.

For more information about Origin metadata see section 5.3.11.3 "def:Origin Element".

```
<!-- ItemGroup Definition (DM) -->
<ItemGroupDef OID="IG.DM"
  Domain="DM"
  Name="DM"
  Repeating="No"
  …
  def:ArchiveLocationID="LF.DM">
  <Description>
    <TranslatedText xml:lang="en">Demographics</TranslatedText>
  </Description>
  …
  <ItemRef ItemOID="IT.STUDYID" OrderNumber="1" Mandatory="Yes" KeySequence="1"/>
  …
  <ItemRef ItemOID="IT.DM.BRTHDTC" OrderNumber="8" Mandatory="No"/>
  …
</ItemGroupDef>

…

<!-- Item Definition: Variable Level (BRTHDTC) -->
<ItemDef OID="IT.DM.BRTHDTC" Name="BRTHDTC" DataType="date" SASFieldName="BRTHDTC">
  <Description>
    <TranslatedText xml:lang="en">Date/Time of Birth</TranslatedText>
  </Description>
  <def:Origin Type="CRF">
    <def:DocumentRef leafID="LF.blankcrf">
      <def:PDFPageRef PageRefs="6" Type="PhysicalRef"/>
    </def:DocumentRef>
  </def:Origin>
</ItemDef>
```

4.2.2.3. Example of Linking an ADaM Variable to a Predecessor Using Origin

This example illustrates the child element *def:Origin* in the *ItemDef* element to identify the source of an ADaM variable (SDTM or other analysis dataset) by using "Predecessor" as the value of the *def:Origin* element's *Type* attribute. In this case the predecessor of the TRTP variable in the ADQSADAS dataset is the TRT01P variable in the ADSL data set.

```
<!-- ItemGroup Definition (ADQSDAS) -->
<ItemGroupDef OID="IG.ADQSADAS"
  Name="ADQSADAS"
  Repeating="Yes"
  IsReferenceData="No"
  Purpose="Analysis"
  …
  def:ArchiveLocationID="LF.ADQSADAS">
  <Description>
    <TranslatedText xml:lang="en">ADAS-Cog Analysis</TranslatedText>
  </Description>
 …
  <ItemRef ItemOID="IT.ADQSADAS.TRTP" OrderNumber="7" Mandatory="No"/>
 …
<ItemGroupDef>

   …

<!-- Item Definition: Variable Level (TRTP) -->
<ItemDef OID="IT.ADQSADAS.TRTP" Name="TRTP" DataType="text" Length="20">
  <Description>
    <TranslatedText xml:lang="en">Planned Treatment</TranslatedText>
  </Description>
  <CodeListRef CodeListOID="CL.ARM"/>
  <def:Origin Type="Predecessor">
    <Description>
      <TranslatedText xml:lang="en">ADSL.TRT01P</TranslatedText>
    </Description>
  </def:Origin>
</ItemDef>
```

4.2.2.4. Example of a Variable with Controlled Terminology Reference

In this example, the AESEV variable uses a NCI/CDISC Controlled Terminology codelist with a set of *CodeListItem* elements inside the surrounding *CodeList* element. Note the references to the Controlled Terminology nci:ExtCodeID codes with the use of the *Alias* element.

Note also that each *EnumeratedItem* element has a *Rank* attribute specified to define the relative value of the CodeListItems.

```
<!-- ItemGroup Definition (AE) -->
<ItemGroupDef OID="IG.AE" Name="AE" Domain="AE"
  …
  <ItemRef ItemOID="IT.AE.AESEV" OrderNumber="10" Mandatory="No"/>
  …
</ItemGroupDef>

<!-- Item Definitions -->
<ItemDef OID="IT.AE.AESEV" Name="AESEV" DataType="text" Length="8"
  SASFieldName="AESEV">
  <Description>
    <TranslatedText xml:lang="en">Severity/Intensity</TranslatedText>
  </Description>
  <CodeListRef CodeListOID="CL.AESEV"/>
  <def:Origin Type="CRF">
    <def:DocumentRef leafID="LF.blankcrf">
      <def:PDFPageRef PageRefs="21" Type="PhysicalRef"/>
    </def:DocumentRef>
  </def:Origin>
</ItemDef>

<CodeList OID="CL.AESEV" Name="Severity/Intensity Scale for Adverse Events"
  DataType="text" SASFormatName="$AESEV">
  <CodeListItem CodedValue="MILD" Rank="1">
    <Decode>
      <TranslatedText xml:lang="en">Grade 1</TranslatedText>
    </Decode>
    <Alias Name="C41338" Context="nci:ExtCodeID"/>
  </CodeListItem>
  <CodeListItem CodedValue="MODERATE" Rank="2">
    <Decode>
      <TranslatedText xml:lang="en">Grade 2</TranslatedText>
    </Decode>
    <Alias Name="C41339" Context="nci:ExtCodeID"/>
  </CodeListItem>
  <CodeListItem CodedValue="SEVERE" Rank="3">
    <Decode>
      <TranslatedText xml:lang="en">Grade 3</TranslatedText>
    </Decode>
    <Alias Name="C41340" Context="nci:ExtCodeID"/>
  </CodeListItem>
  <Alias Name="C66769" Context="nci:ExtCodeID"/>
</CodeList>
```

4.2.2.5. Example of a Derived Variable

This example illustrates the use of the ODM *MethodDef* element to document the algorithm used to derive values for the SESTDTC and SEENDTC variables. For more information about representing computational algorithm metadata data see section 4.6.

```
<!-- Dataset Definition (SE) -->
<ItemGroupDef OID="IG.SE"
  Domain="SE" Name="SE"
  …
  <Description>
    <TranslatedText xml:lang="en">Subject Elements</TranslatedText>
  </Description>
  …
  <ItemRef ItemOID="IT.SE.SESTDTC" OrderNumber="7" Mandatory="Yes" KeySequence="3"
    MethodOID="MT.SESTDTC"/>
  <ItemRef ItemOID="IT.SE.SEENDTC" OrderNumber="8" Mandatory="No" KeySequence="4"
    MethodOID="MT.SEENDTC"/>
  …
</ItemGroupDef>
  …
  <!-- Item Definition: Variable Level (SEENDTC) -->
  <ItemDef OID="IT.SE.SEENDTC" Name="SEENDTC" DataType="date" SASFieldName="SEENDTC">
    <Description>
      <TranslatedText xml:lang="en">End Date/Time of Element</TranslatedText>
    </Description>
    <def:Origin Type="Derived"/>
  </ItemDef>
  …
  <!-- Item Definition: Variable Level (SESTDTC) -->
  <ItemDef OID="IT.SE.SESTDTC" Name="SESTDTC" DataType="date" SASFieldName="SESTDTC">
    <Description>
      <TranslatedText xml:lang="en">Start Date/Time of Element</TranslatedText>
    </Description>
    <def:Origin Type="Derived"/>
  </ItemDef>
  …
  <!-- Method Definitions -->
  <MethodDef OID="MT.SESTDTC" Name="Algorithm to derive SESTDTC" Type="Computation" >
    <Description>
      <TranslatedText xml:lang="en"> If Element = SCREEN, derived from SVSTDTC where VISIT =
        SCREENING or from DS where DSDECOD = 'INFORMED CONSENT', whichever is earliest. If
        Element = EOS, derived from DS where DSCAT = DISPOSITION EVENT. For treatment
        Elements, derived from first EXSTDTC for the element.</TranslatedText>
    </Description>
  </MethodDef>
  <MethodDef OID="MT.SEENDTC" Name="Algorithm to derive SEENDTC" Type="Computation">
    <Description>
      <TranslatedText xml:lang="en"> If Element = SCREEN, derived from SVENDTC where VISIT =
        SCREENING. If Element = EOS, derived from DS where DSCAT = DISPOSITION EVENT or from
        the latest EXENDTC from EX whichever is later. For treatment Elements, derived from
        last EXENDTC for the element. For the complete algorithm see the referenced external
        document.</TranslatedText>
    </Description>
  </MethodDef>
```

4.2.2.6. Example of an ADaM Derived Variable

This example illustrates the use of the ODM *MethodDef* element to document the algorithm used to derive values for the CHG variable in an ADaM Basic Data Structure dataset and the use of the *def:Origin* element to document the Source metadata. For information about representing computational algorithm metadata data see section 4.6.

```
  <!-- ItemGroup Definition (ADQSDAS) -->
  <ItemGroupDef OID="IG.ADQSADAS"
    Name="ADQSADAS"
    Repeating="Yes"
    IsReferenceData="No"
    Purpose="Analysis"
    def:Structure="One record per subject per parameter per analysis visit per
analysis date"
    def:Class="BASIC DATA STRUCTURE"
    def:CommentOID="COM.ADQSADAS"
    def:ArchiveLocationID="LF.ADQSADAS">
    <Description>
      <TranslatedText xml:lang="en">ADAS-Cog Analysis</TranslatedText>
    </Description>
    …
    <ItemRef ItemOID="IT.ADQSADAS.CHG" OrderNumber="29" Mandatory="No"
      MethodOID="MT.ADQSADAS.CHG"/>
    …
    <def:leaf ID="LF.ADQSADAS" xlink:href="adqsadas.xpt">
      <def:title>adqsadas.xpt </def:title>
        </def:leaf>
  </ItemGroupDef>
  …
  <!-- Item Definition: Variable Level (AVISIT) -->
  <ItemDef OID="IT.ADQSADAS.CHG" Name="CHG" DataType="integer" Length="8">
    <Description>
      <TranslatedText xml:lang="en">Change from Baseline</TranslatedText>
    </Description>
    <def:Origin Type="Derived"/>
  </ItemDef>
  …
  <!-- Method Definition: Algorithm description -->
  <MethodDef OID="MT.ADQSADAS.CHG" Name="CM.ADQSADAS.CHG" Type="Computation">
    <Description>
      <TranslatedText xml:lang="en">AVAL - BASE</TranslatedText>
    </Description>
  </MethodDef>
```

4.3. **Controlled Terminology Definitions**

The term "Controlled Terminology" in the context of a study refers to the set of all allowable values across all variables that have finite sets of allowable values in the study. A "Codelist" is a unique subset of the controlled terminology to which one or more variables are subject. Beginning with SDTM Version 1.2, the SDTM-IG requires controlled terminology for many SDTM variables. For some variables, sponsor-specific controlled terminology is recommended. All controlled terminology used in a study must be provided within the Define-XML document. Each codelist referenced by a study item shall be represented in the Define-XML document using a *CodeList* element.

The *CodeList* element can define either an internal or external codelist. Internal codelists include a list of allowable codes and, if applicable, their corresponding decodes. In most cases where controlled terminology is just an enumeration of allowed values, the *EnumeratedItem* element can be used to define the list of values. In cases where it is useful to provide decodes for the coded values, the *CodeListItem* element can be used.

Codelists provided by third parties are specified using an *ExternalCodeList* element that identifies the dictionary by name and version. The *href* attribute contains the URL to the third-party dictionary if one is available. Please note that ItemDefs with date and time related datatypes cannot use codelists, and cannot reference the ISO8601 standard using ExternalCodeLists. The ISO8601 standard is not a controlled terminology, it is an international formatting standard for dates, times, datetimes, and durations.

With few exceptions, coded values for CDISC codelists are provided as upper case text. For variables whose definitions reference *CodeList* elements, the variable data content is expected to **exactly** match a single item in the corresponding CodeList – **including** case.

For codelists with entries that have a numeric significance, this can be described using the *Rank* attribute. For example, a list of enumerated text values including "Low", "Medium", and "High" might include a *Rank* attribute on each with values 1, 2, and 3 respectively. This can be used to override the normal lexical ordering of the entries with one based on numeric significance. The *Rank* attribute should not be used to define a display order.

The *OrderNumber* attribute can be used to define a display order for the items in a codelist.

CodeLists should never include a blank coded value. If an item does not require a value then the *Mandatory* attribute in the *ItemRef* element must be set to "No". Required items have Mandatory set to "Yes" and cannot have blank values.

CDISC codelists can be defined as *extensible*, which means controlled terms may be added to the codelist. Sponsors may add to extensible codelists as long as they are not adding duplicates or synonyms of existing terms. If a CDISC codelist is not extensible, the expectation is that sponsors will use only the published list of terms. If a Define-XML *CodeList* element includes definitions of terms that are not included in the published list, then the *def:ExtendedValue* attribute should be set to "Y" regardless of whether or not the codelist is extensible.

A codelist definition may include one or more *Alias* elements in order to facilitate the identification of the codelist components in an external system. For *CodeList* elements that define the contents of a CDISC codelist, an *Alias* element is used to provide the C-codes that are used to identify codelists and coded terms within the National Cancer Institute's Enterprise Vocabulary System. By convention the Alias *Context* attribute is set to "nci:ExtCodeId" in this scenario.

4.3.1. Examples of Controlled Terminology Definitions

4.3.1.1. Example of a CodeList with EnumeratedItem child elements and Aliases

This example illustrates the use of a CDISC codelist. Aliases are included at both the CodeList and EnumeratedItem levels. At the CodeList level the Alias Context attribute references "nci:ExtCodeID" to indicate the NCI/CDISC SDTM Terminology standards. At the EnumeratedItem level the Alias Context attribute references "nci:ExtCodeID" to indicate a specific term from the NCI/CDISC SDTM Terminology standards.

Note that the location of the Alias element at the CodeList level is after all CodeList *EnumeratedItem* child elements.

```
<CodeList OID="CL.ACN" Name="Action Taken with Study Treatment" DataType="text">
  <EnumeratedItem CodedValue="DOSE NOT CHANGED">
    <Alias Name="C49504" Context="nci:ExtCodeID"/>
  </EnumeratedItem>
  <EnumeratedItem CodedValue="DOSE REDUCED">
    <Alias Name="C49505" Context="nci:ExtCodeID"/>
  </EnumeratedItem>
  <EnumeratedItem CodedValue="DRUG INTERRUPTED">
    <Alias Name="C49501" Context="nci:ExtCodeID"/>
  </EnumeratedItem>
  <EnumeratedItem CodedValue="DRUG WITHDRAWN">
    <Alias Name="C49502" Context="nci:ExtCodeID"/>
  </EnumeratedItem>
  <Alias Name="C66767" Context="nci:ExtCodeID"/>
</CodeList>
```

4.3.1.2. Example of CodeLists for VSTEST and VSTESTCD

The example below shows CodeLists for VSTEST and VSTESTCD variables. The VSTEST CodeList uses EnumeratedItems to store the list of possible values, as listed in the NCI/CDISC SDTM Controlled Terminology documents. The VSTESTCD CodeList uses CodeListItems to reflect the association with VSTEST items.

```
<CodeList OID="CL.VSTEST" Name="Vital Signs Test Name" DataType="text">
  <EnumeratedItem CodedValue="Body Frame Size">
    <Alias Name="C49680" Context="nci:ExtCodeID"/>
  </EnumeratedItem>
  <EnumeratedItem CodedValue="Diastolic Blood Pressure">
    <Alias Name="C25299" Context="nci:ExtCodeID"/>
  </EnumeratedItem>
  <EnumeratedItem CodedValue="Height">
    <Alias Name="C25347" Context="nci:ExtCodeID"/>
  </EnumeratedItem>
  <EnumeratedItem CodedValue="Pulse Rate">
    <Alias Name="C49676" Context="nci:ExtCodeID"/>
  </EnumeratedItem>
  <EnumeratedItem CodedValue="Systolic Blood Pressure">
    <Alias Name="C25298" Context="nci:ExtCodeID"/>
  </EnumeratedItem>
  <EnumeratedItem CodedValue="Weight">
    <Alias Name="C25208" Context="nci:ExtCodeID"/>
  </EnumeratedItem>
  <Alias Name="C67153" Context="nci:ExtCodeID"/>
</CodeList>

<CodeList OID="CL.VSTESTCD" Name="Vital Signs Test Code" DataType="text"
  SASFormatName="$VSTESTC">
  <CodeListItem CodedValue="DIABP">
    <Decode>
      <TranslatedText xml:lang="en">Diastolic Blood Pressure</TranslatedText>
    </Decode>
    <Alias Name="C25299" Context="nci:ExtCodeID"/>
  </CodeListItem>
  <CodeListItem CodedValue="FRMSIZE">
    <Decode>
      <TranslatedText xml:lang="en">Body Frame Size</TranslatedText>
    </Decode>
    <Alias Name="C49680" Context="nci:ExtCodeID"/>
  </CodeListItem>
  <CodeListItem CodedValue="HEIGHT">
    <Decode>
      <TranslatedText xml:lang="en">Height</TranslatedText>
    </Decode>
    <Alias Name="C25347" Context="nci:ExtCodeID"/>
  </CodeListItem>
  <CodeListItem CodedValue="PULSE">
    <Decode>
      <TranslatedText xml:lang="en">Pulse Rate</TranslatedText>
    </Decode>
    <Alias Name="C49676" Context="nci:ExtCodeID"/>
  </CodeListItem>
  <CodeListItem CodedValue="SYSBP">
    <Decode>
      <TranslatedText xml:lang="en">Systolic Blood Pressure</TranslatedText>
    </Decode>
    <Alias Name="C25298" Context="nci:ExtCodeID"/>
  </CodeListItem>
  <CodeListItem CodedValue="WEIGHT">
    <Decode>
      <TranslatedText xml:lang="en">Weight</TranslatedText>
    </Decode>
    <Alias Name="C25208" Context="nci:ExtCodeID"/>
  </CodeListItem>
  <Alias Name="C66741" Context="nci:ExtCodeID"/>
</CodeList>
```

4.3.1.3. Example of a CodeList with CodeListItem Child elements

In this example, controlled terminology for the SDTM variable ARMCD is defined. This terminology requires decoded values to facilitate data interpretation, so CodeList includes *CodeListItem* elements with a *Decode* element. Each *Decode* element, in turn, includes a *TranslatedText* element that contains the decoded value.

Note also that the OrderNumber attribute allows to display the elements in a different sequence than the order listed in the *define.xml* document.

```
<CodeList OID="CL.ARMCD" Name="Planned Arm Code" DataType="text"
  SASFormatName="$ARMCD">
  <CodeListItem CodedValue="PLACEBO" OrderNumber="3">
    <Decode><TranslatedText xml:lang="en">Placebo</TranslatedText></Decode>
  </CodeListItem>
  <CodeListItem CodedValue="SCRNFAIL" OrderNumber="4">
    <Decode><TranslatedText xml:lang="en">Screen Failure</TranslatedText></Decode>
  </CodeListItem>
  <CodeListItem CodedValue="WONDER10" OrderNumber="1">
    <Decode><TranslatedText xml:lang="en">Miracle Drug 10 mg</TranslatedText></Decode>
  </CodeListItem>
  <CodeListItem CodedValue="WONDER20" OrderNumber="2">
    <Decode><TranslatedText xml:lang="en">Miracle Drug 20 mg</TranslatedText></Decode>
  </CodeListItem>
</CodeList>
```

4.3.1.4. Example of a CodeList using Rank attribute

In this example, controlled terminology for the SDTM variable AESEV is defined using the SDTM Codelist AESEV. The *Rank* attribute is provided for each EnumeratedItem to indicate temporal order in analysis computations.

```
<CodeList OID="CL.AESEV" Name="Severity/Intensity Scale for Adverse Events"
  DataType="text" SASFormatName="$AESEV">
  <CodeListItem CodedValue="MILD" Rank="1">
    <Decode>
      <TranslatedText xml:lang="en">Grade 1</TranslatedText>
    </Decode>
    <Alias Name="C41338" Context="nci:ExtCodeID"/>
  </CodeListItem>
  <CodeListItem CodedValue="MODERATE" Rank="2">
    <Decode>
      <TranslatedText xml:lang="en">Grade 2</TranslatedText>
    </Decode>
    <Alias Name="C41339" Context="nci:ExtCodeID"/>
  </CodeListItem>
  <CodeListItem CodedValue="SEVERE" Rank="3">
    <Decode>
      <TranslatedText xml:lang="en">Grade 3</TranslatedText>
    </Decode>
    <Alias Name="C41340" Context="nci:ExtCodeID"/>
  </CodeListItem>
  <Alias Name="C66769" Context="nci:ExtCodeID"/>
</CodeList>
```

4.3.1.5. Example of a Codelist with ExternalCodeList child element

In this example, controlled terminology for the SDTM variable AEDECOD is defined using a third party dictionary, MedDRA. The contents for this terminology must be licensed through the Medical Dictionary for Regulatory Activities Maintenance and Support Services Organization, but to facilitate interpretation of the clinical data, a *CodeList* element is defined using an *ExternalCodeList* element to specify the dictionary name and version.

```
<ItemDef OID="IT.AE.AEDECOD" Name="AEDECOD" DataType="text" Length="18"
  SASFieldName="AEDECOD">
  <Description>
    <TranslatedText xml:lang="en">Dictionary-Derived Term</TranslatedText>
  </Description>
  <CodeListRef CodeListOID="CL.AEDICT_F"/>
  <def:Origin Type="Assigned"/>
</ItemDef>

<CodeList OID="CL.AEDICT_F" Name="Adverse Event Dictionary" DataType="text">
  <ExternalCodeList Dictionary="MedDRA" Version="14.0"/>
</CodeList>
```

4.3.1.6. Example of a CDISC Controlled Terminology with an Extended Item

In this example, a CodeListItem ("SUBJINIT") has been added by the sponsor to extend external controlled terminology. Note that the last *CodeListItem* element does not include an *Alias* element since it represents a value added to an extensible CDISC Controlled Terminology. The final *Alias* element identifies the CDISC Controlled Terminology for "Subject Characteristic Test Code (SCTESTCD)".

```
<CodeList OID="CL.SCTESTCD" Name="Subject Characteristic Code (SCTESTCD)"
  DataType="text" SASFormatName="$SCTESTC">
  <CodeListItem CodedValue="EDLEVEL">
    <Decode>
      <TranslatedText xml:lang="en">Education Level</TranslatedText>
    </Decode>
    <Alias Name="C17953" Context="nci:ExtCodeID"/>
  </CodeListItem>
  <CodeListItem CodedValue="MARISTAT">
    <Decode>
      <TranslatedText xml:lang="en">Marital Status</TranslatedText>
    </Decode>
    <Alias Name="C25188" Context="nci:ExtCodeID"/>
  </CodeListItem>
  <CodeListItem CodedValue="SUBJINIT" def:ExtendedValue="Yes">
    <Decode>
      <TranslatedText xml:lang="en">Subject Initials</TranslatedText>
    </Decode>
  </CodeListItem>
  <Alias Name="C74559" Context="nci:ExtCodeID"/>
</CodeList>
```

4.4. **Value Level Metadata Definitions**

The normalized data structure used by datasets based on the SDTM, SEND and ADaM models (generally one record per subject per test code per visit or observation) provides an efficient method for transmitting information. However, there are cases where the dataset variable metadata does not provide sufficient detail to support data review and analysis. In these cases Value Level Metadata should be provided in the Define-XML document. Value Level Metadata enables the specification of the metadata of a variable under conditions involving one or more other dataset variables. The definition of a variable for a specific condition is known as *Value Level Metadata*.

Value Level Metadata is specified in the Define-XML using *def:ValueListDef* elements (see section 5.3.8 for a detailed specification). Value List definitions (or valuelists) are linked with variable definitions by including a *def:ValueListRef* element in the *ItemDef* element that defines the Variable. See section 5.3.11.2 for a detailed specification of the *def:ValueListRef* element.

Note: In Define-XML v1.0.0, a valuelist definition could be attached to any variable, but in conventional usage was attached only to the --TESTCD Variable in the Findings Class or the SUPP--.QNAM variable in the special class SUPPQUAL. It was assumed to describe the contents of the --ORRES variable for that Test Code and the SUPP--.QVAL variable for that SUPP--.QNAM respectively. In Define-XML v2.0.0 valuelists should be attached to the variable being defined so that any number of variables can be defined in detail.

Value Level Metadata should be provided when there is a need to describe differing metadata attributes for subsets of cells within a column. It is most often used on SDTM Findings domains to provide definitions for Variables such as --ORRES, --ORRESU, --STRES, --STRESU that are specific to each test code (value of --TESTCD). It is not required for Findings domains where the results have the same characteristics in all records, such as IE domains. In ADaM, value level metadata often describes AVAL or AVALC in BDS data structures based on values of PARAMCD.

Value Level Metadata can also be used on other types of SDTM domains. For example, in a DS domain, Value Level Metadata might define the codelists for DSTERM and DSDECOD for each value of DSCAT.

Value Level Metadata should be applied when it provides information useful for interpreting study data. It need not be applied in all cases.

As an example, the --TEST variable could either be specified by a single variable with a codelist containing all the test names, or it could have Value Level Metadata specifying exactly which test name is appropriate for each --TESTCD. Both of these approaches are valid but the Value Level Metadata approach is more complicated and may not provide any information that will benefit a consumer of the data. It is left to the discretion of the creator when it is useful to provide Value Level Metadata and when it is not. The creator should speak to the consumer if they would like further guidance.

When deciding whether or not to define Value Level Metadata, a good rule of thumb is that if providing Value Level Metadata for a variable would mean each value has a codelist containing only one CodeList Item, it is probably more appropriate to define a codelist rather than a valuelist. This is the case with --TEST and --TESTCD Variables.

Similarly, if the values of the *DataType* and *Length* attributes of each value are the same and there is no codelist involved, it is not necessary to provide Value Level Metadata. Value Level

Metadata can still be provided if desired, for example to specify a different Origin for a variable for each visit it appears at, however this level of granularity is not required.

Variable Level Metadata applies to all cells in a column within a table unless overridden by Value Level Metadata. This applies to individual properties of a column such as Length or codelist, so if Variable Level Metadata for a column specifies a Length of "8" but Value Level Metadata does not give a Length, the value of "8" from the Variable Level Metadata is used. As such, it is not necessary to supply Value Level Metadata for properties that do not change at the Value level.

Note that there is no requirement to provide a codelist on a parent variable if codelists are provided at the Value level, though one can be provided if desired. If the user provides codelists at both levels, it is expected that a codelist defined at the Variable level will be a superset of all codelists specified at the Value level for the specified Variable.

Value level definitions must be compatible with the parent variable definition. For example, no value-level length can exceed the length of the parent variable.

The following Item may be used to define a VSORRES variable:

```
<ItemDef OID="IT.VS.VSORRES" Name="VSORRES" DataType="text" Length="30"
  SASFieldName="VSORRES">
  <Description>
    <TranslatedText xml:lang="en">Result or Finding in Original Units</TranslatedText>
  </Description>
  <def:Origin Type="CRF">
    <def:DocumentRef leafID="LF.blankcrf">
      <def:PDFPageRef PageRefs="11" Type="PhysicalRef"/>
    </def:DocumentRef>
  </def:Origin>
  <def:ValueListRef ValueListOID="VL.VS.VSORRES"/>
</ItemDef>
```

The following Item may be used to define the DIABP original result's Value within the VSORRES column:

```
<ItemDef OID="IT.VS.VSORRES.DIABP" Name="DIABP" DataType="integer" Length="2"
  SASFieldName="DIABP">
  <Description>
    <TranslatedText xml:lang="en">Diastolic Blood Pressure</TranslatedText>
  </Description>
</ItemDef>
```

In this example VSORRES has a length of 30 and can hold any text value, while VSORRES values for DIABP records must be integers of length no greater than 2.

4.4.1. Where Clauses for Value Level Metadata

Where Clauses are used to describe the conditions under which the definition of a Value applies in a machine-readable form. Each Value definition may have a Where Clause attached to it to describe when that Value applies. The Where Clause mechanism allows the definition of slices. Slices are subsets of a dataset that typically include a subset of the dataset rows that share similar metadata.

Where Clauses define a condition by using one or more Range Checks. Where there are multiple Range Checks the condition is defined by the logical AND of all the Range Checks.

This allows the construction of compound Where Clauses, e.g.:

```
<!--  Where Clause definitions for:
      Where VSTESTCD = 'SYSBP' and VSPOS = 'SITTING' -->
<def:WhereClauseDef OID="WC.VS.VSTESTCD.SYSBP.VS.VSPOS.SITTING">
  <RangeCheck SoftHard="Soft" def:ItemOID="IT.VS.VSTESTCD" Comparator="EQ">
    <CheckValue>SYSBP</CheckValue>
  </RangeCheck>
  <RangeCheck SoftHard="Soft" def:ItemOID="IT.VS.VSPOS" Comparator="EQ">
    <CheckValue>SITTING</CheckValue>
  </RangeCheck>
</def:WhereClauseDef>
```

```
<!--  Where Clause definitions for:
      Where VSTESTCD = 'WEIGHT' and COUNTRY WITH METRIC SYSTEM  -->
<def:WhereClauseDef OID="WC.VS.VSTESTCD.WEIGHT.[DM].COUNTRY.CMETRIC"
  def:CommentOID="COM.SUBJECTDATA-JOIN-DM">
  <RangeCheck SoftHard="Soft" def:ItemOID="IT.VS.VSTESTCD" Comparator="EQ">
    <CheckValue>WEIGHT</CheckValue>
  </RangeCheck>
  <RangeCheck SoftHard="Soft" def:ItemOID="IT.DM.COUNTRY" Comparator="IN">
    <CheckValue>CAN</CheckValue>
    <CheckValue>MEX</CheckValue>
  </RangeCheck>
</def:WhereClauseDef>
```

Where Clauses can only contain references to:

1. Variables in the current dataset
2. Variables in other subject-level datasets within the same metadata version

In case #2, the implied dataset join should be documented in a comment attached to the WhereClauseDef.

CAVEAT: In the current specification, there is no mechanism for machine-readable join specifications. Therefore, the Comment functionality is intended to document the implied join.

4.4.2. Examples of Value Level Metadata Definitions

It is important to understand that only one XML representation of Value Level Metadata exists. However, to facilitate understanding of these concepts, this document discusses Value Level Metadata in terms of both *Value Lists* and *Slices*. Value Lists and Slices are simply two different ways of visualizing the same underlying metadata. Value Lists show the definitions for a single variable for each condition while Slices show the definitions for a whole Domain for a given condition (i.e. Where clause). More details about the distinction between these two visualizations are illustrated in the examples in Appendix 2.

4.4.2.1. Example of Value Level Metadata – Vital Signs Domain

```
<!--  Value Level Metadata definitions  -->
  <def:ValueListDef OID="VL.VS.VSORRES">
    <ItemRef ItemOID="IT.VS.VSORRES.DIABP" OrderNumber="1" Mandatory="Yes">
      <def:WhereClauseRef WhereClauseOID="WC.VS.VSTESTCD.DIABP"/>
    </ItemRef>
    …
    <ItemRef ItemOID="IT.VS.VSORRES.HEIGHT" OrderNumber="3" Mandatory="Yes">
      <def:WhereClauseRef WhereClauseOID="WC.VS.VSTESTCD.HEIGHT"/>
    </ItemRef>
    …
    <ItemRef ItemOID="IT.VS.VSORRES.WEIGHT" OrderNumber="6" Mandatory="Yes">
      <def:WhereClauseRef WhereClauseOID="WC.VS.VSTESTCD.WEIGHT"/>
    </ItemRef>
  </def:ValueListDef>
  <def:ValueListDef OID="VL.VS.VSORRESU">
    <ItemRef ItemOID="IT.VS.VSORRESU.HEIGHT.DM.COUNTRY.CMETRIC"
      OrderNumber="1" Mandatory="Yes">
      <def:WhereClauseRef
        WhereClauseOID="WC.VS.VSTESTCD.HEIGHT.[DM].COUNTRY.CMETRIC"/>
    </ItemRef>
    <ItemRef ItemOID="IT.VS.VSORRESU.HEIGHT.DM.COUNTRY.CNMETRIC
      OrderNumber="2" Mandatory="Yes">
      <def:WhereClauseRef
        WhereClauseOID="WC.VS.VSTESTCD.HEIGHT.[DM].COUNTRY.CNMETRIC"/>
    </ItemRef>
    <ItemRef ItemOID="IT.VS.VSORRESU.WEIGHT.DM.COUNTRY.CMETRIC"
      OrderNumber="3" Mandatory="Yes">
      <def:WhereClauseRef
        WhereClauseOID="WC.VS.VSTESTCD.WEIGHT.[DM].COUNTRY.CMETRIC"/>
    </ItemRef>
    <ItemRef ItemOID="IT.VS.VSORRESU.WEIGHT.DM.COUNTRY.CNMETRIC"
      OrderNumber="4" Mandatory="Yes">
      <def:WhereClauseRef
        WhereClauseOID="WC.VS.VSTESTCD.WEIGHT.[DM].COUNTRY.CNMETRIC"/>
    </ItemRef>
  </def:ValueListDef>
```

```
<!--  Item definition for VSORRES -->
<ItemDef OID="IT.VS.VSORRES" Name="VSORRES" DataType="text" Length="200"
  SASFieldName="VSORRES">
  <Description>
    <TranslatedText xml:lang="en">Result or Finding in Original Units</TranslatedText>
  </Description>
  <def:ValueListRef ValueListOID="VL.VS.VSORRES"/>
</ItemDef>
<!--  Item definition for VSORRESU -->
<ItemDef OID="IT.VS.VSORRESU" Name="VSORRESU" DataType="text" Length="9"
  SASFieldName="VSORRESU">
  <Description>
    <TranslatedText xml:lang="en">Standard Units</TranslatedText>
  </Description>
  <def:ValueListRef ValueListOID="VL.VS.VSORRESU"/>
</ItemDef>
...
<ItemDef OID="IT.VS.VSORRES.DIABP" Name="DIABP" DataType="integer" Length="2"
  SASFieldName="DIABP">
  <Description>
    <TranslatedText xml:lang="en">Diastolic Blood Pressure</TranslatedText>
  </Description>
</ItemDef>
...
<ItemDef OID="IT.VS.VSORRESU.HEIGHT.DM.COUNTRY.CMETRIC" Name="HEIGHT" DataType="float"
  Length="5" SASFieldName="HEIGHT">
  <Description>
    <TranslatedText xml:lang="en">Height</TranslatedText>
  </Description>
  <CodeListRef CodeListOID="CL.UH_MC"/>
</ItemDef>
<ItemDef OID="IT.VS.VSORRESU.HEIGHT.DM.COUNTRY.CNMETRIC" Name="HEIGHT" DataType="float"
   Length="5" SASFieldName="HEIGHT">
  <Description>
    <TranslatedText xml:lang="en">Height</TranslatedText>
  </Description>
  <CodeListRef CodeListOID="CL.UH_NMC"/>
</ItemDef>

<!--  Item definition for VSSTRESU -->
<ItemDef OID="IT.VS.VSSTRESU" Name="VSSTRESU" DataType="text" Length="9"
SASFieldName="VSSTRESU">
  <Description>
    <TranslatedText xml:lang="en">Standard Units</TranslatedText>
  </Description>
  <CodeListRef CodeListOID="CL.VSRESU"/>
  <def:Origin Type="Derived"/>
</ItemDef>
```

In this example, Value Level Metadata is provided for the VS domain. The VS domain is a Findings domain and so the attributes of variables VSORRES, VSORRESU, VSSTRES, VSSTRESU and other variables can have different values depending on the value of the VSTESTCD and values of other variables. Therefore, a description of all the different possible values for those variables at a more granular level is appropriate. This is not a complete example and only variables VSORRES and VSORRESU are shown with a Value List attached to them. In contrast, variable VSSTRESU is illustrated only with a codelist that covers all the possible values.

VSORRES has different Value definitions. The example shows the definition of VSORRES when VSTESTCD='DIAPBP'.

VSOSRESU also has different possible value definitions, such as one for VSTESTCD='HEIGHT' in countries that use the Metric system and one for VSTESTCD='HEIGHT' in countries that do not use the Metric system.

4.4.2.2. Example of Where Clause Metadata – Vital Signs Domain

```
<def:WhereClauseDef OID="WC.VS.VSTESTCD.HEIGHT.[DM].COUNTRY.CMETRIC"
    def:CommentOID="COM.SUBJECTDATA-JOIN-DM">
  <RangeCheck SoftHard="Soft" def:ItemOID="IT.VS.VSTESTCD" Comparator="EQ">
    <CheckValue>HEIGHT</CheckValue>
  </RangeCheck>
  <RangeCheck SoftHard="Soft" def:ItemOID="IT.DM.COUNTRY" Comparator="IN">
    <CheckValue>CAN</CheckValue>
    <CheckValue>MEX</CheckValue>
  </RangeCheck>
</def:WhereClauseDef>
<def:WhereClauseDef OID="WC.VS.VSTESTCD.HEIGHT.[DM].COUNTRY.CNMETRIC"
    def:CommentOID="COM.SUBJECTDATA-JOIN-DM">
  <RangeCheck SoftHard="Soft" def:ItemOID="IT.VS.VSTESTCD" Comparator="EQ">
    <CheckValue>HEIGHT</CheckValue>
  </RangeCheck>
  <RangeCheck SoftHard="Soft" def:ItemOID="IT.DM.COUNTRY" Comparator="EQ">
    <CheckValue>USA</CheckValue>
  </RangeCheck>
</def:WhereClauseDef>

<!--  Where Clause definitions for:  Where VSTESTCD = 'DIABP'  -->
<def:WhereClauseDef OID="WC.VS.VSTESTCD.DIABP ">
  <RangeCheck SoftHard="Soft" def:ItemOID="IT.VS.VSTESTCD" Comparator="EQ">
    <CheckValue>DIABP</CheckValue>
  </RangeCheck>
</def:WhereClauseDef>

<!--  Documentation to join a subject-level dataset with the Demographics dataset -->
<def:CommentDef OID="COM.SUBJECTDATA-JOIN-DM">
  <Description>
    <TranslatedText xml:lang="en">Join any Subject Level dataset with the Demographics
     dataset based on [IG.datasetname]IT.USUBJID = [IG.DM]IT.USUBJID, assuming
     'IG.datasetname' is the OID of the ItemGroupDef that defines the subject-level
     dataset to be joined with the Demographics dataset.</TranslatedText>
  </Description>
</def:CommentDef>
```

In this example there are three Where Clauses shown. They define three slices or conditions for the VS domain:

- Where VSTESTCD = 'HEIGHT' and COUNTRY IN ['CAN', 'MEX']
- Where VSTESTCD = 'HEIGHT' and COUNTRY IN ['USA']
- Where VSTESTCD = 'DIABP'

As shown in the example in section 4.4.2.1, each value definition requires its own Where Clause.

For VSORRES, a single DIABP Value definition is used regardless of the COUNTRY, so that the Value definition Where Clause has only a single condition, VSTESTCD='DIABP'.

For VSORRESU, there are two possible definitions for HEIGHT depending on the value of COUNTRY. For this reason, a Value definition and corresponding Where Clause is provided for each condition.

4.4.2.3. Example of Value Level Metadata -- SUPPQUAL

This example illustrates Value Level definitions for variable QVAL in the SUPPLB and SUPPQS datasets as supplemental or non-standard variables for the LB and QS domains.

```
<def:ValueListDef OID="VL.SUPPLB.QVAL">
  <ItemRef ItemOID="IT.SUPPLB.QVAL.LBCLSIG" OrderNumber="1" Mandatory="No"
    MethodOID="MT.CLSIG">
    <def:WhereClauseRef WhereClauseOID="WC.SUPPLB.QNAM.LBCLSIG"/>
  </ItemRef>
</def:ValueListDef>
<def:ValueListDef OID="VL.SUPPQS.QVAL">
  <ItemRef ItemOID="IT.SUPPQS.QVAL.RTRINIT" OrderNumber="1" Mandatory="No">
    <def:WhereClauseRef WhereClauseOID="WC.SUPPQS.QNAM.RTRINIT"/>
  </ItemRef>
</def:ValueListDef>

<ItemDef OID="IT.SUPPLB.QVAL.LBCLSIG" Name="LBCLSIG" DataType="text" Length="1"
  SASFieldName="LBCLSIG">
  <Description>
    <TranslatedText xml:lang="en">Clinically Significant</TranslatedText>
  </Description>
  <CodeListRef CodeListOID="CL.NY"/>
  <def:Origin Type="Derived"/>
</ItemDef>

<ItemDef OID="IT.SUPPQS.QVAL.RTRINIT" Name="RTRINIT" DataType="text" Length="3"
  SASFieldName="RTRINIT"
  def:CommentOID="COM.SUPPQS.QVAL.RTRINIT">
  <Description>
    <TranslatedText xml:lang="en">Rater Initials</TranslatedText>
  </Description>
  <def:Origin Type="CRF">
    <def:DocumentRef leafID="LF.blankcrf">
      <def:PDFPageRef PageRefs="13 14 17" Type="PhysicalRef"/>
    </def:DocumentRef>
  </def:Origin>
</ItemDef>
```

Note the definition of the Value List is for the variable QVAL and not for the variable QNAM.

4.4.2.4. Example of ADaM Parameter Level Metadata

This example illustrates the use of the *def:ValueListDef* with a reference to a def:WhereClauseRef used to define metadata for AVAL based on the value of the PARAMCD variable in an ADaM Basic Data Structure dataset.
Note that the only two parameter level definitions are required in this case because the metadata is the same for all parameter values except one.
The example illustrates the concepts of linking between a definition of an object (black arrows) and references to it (red arrows).

```
<def:ValueListDef OID="VL.ADQSADAS.AVAL">
  <ItemRef ItemOID="IT.ADQSADAS.AVAL.ACITM01-ACITM14"
    Mandatory="No" MethodOID="MT.ADQSADAS.AVAL.ACITM01-ACITM14">
    <def:WhereClauseRef WhereClauseOID="WC.ADQSADAS.AVAL.ACITM01-ACITM14"/>
  </ItemRef>
  <ItemRef ItemOID="ADQSADAS.AVAL.ACTOT"
    Mandatory="No" MethodOID="MT.ADQSADAS.AVAL.ACTOT">
    <def:WhereClauseRef WhereClauseOID="WC.ADQSADAS.AVAL.ACTOT"/>
  </ItemRef>
</def:ValueListDef>
. . .

<def:WhereClauseDef OID="WC.ADQSADAS.AVAL.ACITM01-ACITM14">
  <RangeCheck Comparator="IN" SoftHard="Soft" def:ItemOID="IT.ADQSADAS.PARAMCD">
    <CheckValue>ACITM01</CheckValue>
    <CheckValue>ACITM02</CheckValue>
    <CheckValue>ACITM03</CheckValue>
    <CheckValue>ACITM04</CheckValue>
    <CheckValue>ACITM05</CheckValue>
    <CheckValue>ACITM06</CheckValue>
    <CheckValue>ACITM07</CheckValue>
    <CheckValue>ACITM08</CheckValue>
    <CheckValue>ACITM09</CheckValue>
    <CheckValue>ACITM10</CheckValue>
    <CheckValue>ACITM11</CheckValue>
    <CheckValue>ACITM12</CheckValue>
    <CheckValue>ACITM13</CheckValue>
    <CheckValue>ACITM14</CheckValue>
  </RangeCheck>
</def:WhereClauseDef>

<def:WhereClauseDef OID="WC.ADQSADAS.AVAL.ACTOT">
  <RangeCheck Comparator="EQ" SoftHard="Soft" def:ItemOID="IT.ADQSADAS.PARAMCD">
    <CheckValue>ACTOT</CheckValue>
  </RangeCheck>
</def:WhereClauseDef>

. . .

<!-- Item Definition: Variable Level (AVAL) -->
<ItemDef OID="IT.ADQSADAS.AVAL" Name="AVAL" SASFieldName="AVAL" DataType="integer" Length="8">
  <Description>
    <TranslatedText xml:lang="en">Analysis Value</TranslatedText>
  </Description>
  <def:ValueListRef ValueListOID="VL.ADQSADAS.AVAL"/>
</ItemDef>

<ItemDef OID="IT.ADQSADAS.AVAL.ACITM01-ACITM14" Name="AVAL" DataType="integer" Length="8">
  <Description>
    <TranslatedText xml:lang="en">Analysis Value</TranslatedText>
  </Description>
  <def:Origin Type="Derived"/>
</ItemDef>

<ItemDef OID="IT.ADQSADAS.AVAL.ACTOT" Name="AVAL" DataType="integer" Length="8">
  <Description>
    <TranslatedText xml:lang="en">Analysis Value</TranslatedText>
  </Description>
  <def:Origin Type="Derived"/>
</ItemDef>

<ItemDef OID="IT.ADQSADAS.PARAMCD" Name="PARAMCD" SASFieldName="APARAMCD"
  DataType="text" Length="8">
  <Description>
    <TranslatedText xml:lang="en">Parameter Code</TranslatedText>
  </Description>
  <CodeListRef CodeListOID="CL.PARAMCD_ADQSADAS"/>
  <def:Origin Type="Assigned"/>
</ItemDef>
```

4.5. **Links to Supporting Documents**

Two types of external documents, Annotated Case Report Forms (CRF) and Supplemental Documents are often provided along with study datasets in a regulatory submission. Information about these documents is typically provided in the Define-XML in order to facilitate the review of study data.

If an Annotated CRF is provided the Define-XML should include a *def:AnnotatedCRF* element. The *def:AnnotatedCRF* element references *def:leaf* elements which identify the location of the external files relative to the folder where the Define-XML is located. Details of how to specify the *def:AnnotatedCRF* element are in section 5.3.6. Details for specification of the *def:leaf* element are in section 5.3.15.

Some sponsors provide supplemental data definition information as an appendix to the Data Definition Document. There are a multitude of reasons why additional or supplemental information is provided. Chief among them is the sponsor's desire to provide regulatory reviewers with further explanations or descriptions of the variables contained within the datasets. For example, some clinical algorithms in the Statistical Analysis Plan (SAP) are complex and may require additional explanation. These are best described in a flow chart or in some other graphical depiction. Additionally, the Comment column (within the Data Definition Table) may not be large enough to adequately explain the variable derivation and/or its usage. In these situations, sponsors often include an appendix section or external file that contains this additional information. The information, however, should not be redundant with any other document contained within the submission (e.g., eCTD Section 16.1.9 contains the "Documentation of statistical methods and interim analysis plans").

If a supplemental document is provided, the Define-XML should include a *def:SupplementalDoc* element. The *def:SupplementalDoc* element references *def:leaf* elements that identify the location of the external files relative to the folder where the Define-XML is located. Details of how to specify the *def:SupplementalDoc* element are in section 5.3.7.

Annotated CRF page numbers may be included in an ItemDef's *def:Origin* element to provide traceability between the data collection CRF and submission dataset variables. See section 5.3.11.3 for details of the *def:Origin* element and section 5.3.6.1.1 for details of the *def:PDFPageRef* element.

4.5.1. Examples of Links to Supporting Documents

4.5.1.1. Example Annotated CRF Reference

```
<def:AnnotatedCRF>
  <def:DocumentRef leafID="LF.blankcrf"/>
</def:AnnotatedCRF>

<def:leaf ID="LF.blankcrf" xlink:href="blankcrf.pdf">
  <def:title>Annotated Case Report Form</def:title>
</def:leaf>
```

The *def:AnnotatedCRF* element contains one *def:DocumentRef* per Annotated CRF, each referencing a *def:leaf* element which in turn contains XLink information linking to the document that, in this case, is located in the same folder as the *define.xml* file.

4.5.1.2. Example Supplemental Documentation Reference

```
<def:SupplementalDoc>
  <def:DocumentRef leafID="LF.ReviewersGuide"/>
  <def:DocumentRef leafID="LF.ComplexAlgorithms"/>
</def:SupplementalDoc>

<def:leaf ID="LF.ReviewersGuide" xlink:href="reviewersguide.pdf">
  <def:title>Reviewers Guide</def:title>
</def:leaf>

<def:leaf ID="LF.ComplexAlgorithms" xlink:href="complexalgorithms.pdf">
  <def:title>Complex Algorithms</def:title>
</def:leaf>
```

The *def:SupplementalDoc* element contains one *def:DocumentRef* per Supplemental Document, each referencing a *def:leaf* element which in turn contains XLink information linking to the document.

4.6. Computational Method Definitions

The *MethodDef* element describes the algorithms used to generate values for variables defined as derived. See section 5.3.13 for details of the *MethodDef* element. The algorithm is linked to a variable or value using the *MethodOID* attribute in the corresponding *ItemRef* element (see section 5.3.8.1).

To enhance traceability users are encouraged to provide descriptions that include accurate and consistent references to source variables and derivations.

For cases where the algorithm description is longer than a few lines, the MethodDef can include a *def:DocumentRef* element to link to a section in a supplemental document containing the additional details. See section 5.3.6.1 for details of the *def:DocumentRef* element.

A MethodDef can also link to a text file that contains the specific programming code to be executed. However, the external location has to be agreed by sender and consumer.

Note that for regulatory submissions to the FDA, the locations specified have to conform to locations allowed in the eCTD and the Study Data Specifications. See section 2.2 References.

4.6.1. Examples of Computational Method Definition

4.6.1.1. Example of Short Method Definition

```
<MethodDef OID="MT.VSDY" Name="Algorithm to derive VSDY" Type="Computation">
  <Description>
    <TranslatedText xml:lang="en">VSDY = VSDTC-RFSTDTC+1 if VSDTC is on or after
      RFSTDTC. VSDTC - RFSTDTC if VSDTC precedes RFSTDTC.</TranslatedText>
  </Description>
</MethodDef>

<MethodDef OID="MT.SESTDTC" Name="Algorithm to derive SESTDTC" Type="Computation">
  <Description>
    <TranslatedText xml:lang="en">If Element = SCREEN, derived from SVSTDTC where
      VISIT = SCREENING or from DS where DSDECOD = 'INFORMED CONSENT', whichever is
      earliest. If Element = EOS, derived from DS where DSCAT = DISPOSITION EVENT.
      For treatment Elements, derived from first EXSTDTC for the element.
    </TranslatedText>
  </Description>
</MethodDef>
```

4.6.1.2. Example of External Method Definition

```
<!-- Method Definition: Algorithm included or expanded in an external file -->
<MethodDef OID="MT.EGDRVFL" Name="Algorithm to derive EGDRVFL" Type="Computation">
  <Description>
    <TranslatedText xml:lang="en">EGDRVFL = "Y" for derived EGTESTCDs QTCB and QTCF.
      Null otherwise. </TranslatedText>
  </Description>
  <def:DocumentRef leafID="LF.ComplexAlgorithms">
    <def:PDFPageRef PageRefs="EG" Type="NamedDestination"/>
  </def:DocumentRef>
</MethodDef>

<def:leaf ID="LF.ComplexAlgorithms" xlink:href="complexalgorithms.pdf">
  <def:title>Complex Algorithms</def:title>
</def:leaf>
```

This example uses a Named Destination which lets you point to a specific place in a PDF document. Named Destinations have to be created within the PDF document to be able to link to them with a hyperlink.

4.6.1.3. Example of Method Definition with Programming Code Reference

```
<MethodDef OID="MT.QTCB" Name="Algorithm to derive QTCB" Type="Computation">
  <Description>
    <TranslatedText xml:lang="en">QTcB = QT interval / square root of
      (60 / heart rate). For the complete algorithm see the referenced
       external document.
    </TranslatedText>
  </Description>
  <def:DocumentRef leafID="LF.CODE.001"/>
</MethodDef>

<def:leaf ID="LF.CODE.001" xlink:href="../programs/QTCB-computation-sas.txt">
  <def:title>QTcB-Bazett's Correction Formula</def:title>
</def:leaf>
```

This example links to a file with SAS programming code. Note that at the time of writing this document, the path in the xlink:href is not listed as applicable to SDTM in the context of a Regulatory Submission to the FDA. The example is provided only to illustrate the functionality.

4.6.1.4. Example of Method Definition with FormalExpression included

```
<MethodDef OID="MT.USUBJID" Name="Algorithm to derive USUBJID" Type="Computation">
  <Description>
    <TranslatedText xml:lang="en">Concatenation of STUDYID and SUBJID
    </TranslatedText>
  </Description>
  <FormalExpression Context="SAS 9.0 or later, as part of a data step assignment or
   proc sql select and update statements.">
    catx(".",STUDYID,SUBJID)
  </FormalExpression>
</MethodDef>
```

This example illustrates a Method definition that includes a FormalExpression in SAS programming code.

4.7. Comment Definitions

Define-XML allows the definition of Comments at both the dataset and the variable and value levels.

The mechanism allows referencing short comments self-contained in the Define-XML document or long comments referenced in external documents. For comments in external documents, the reference can include specific pages within the document.

Comments are not intended to replace a properly defined computational algorithm, which is expected for derived variables.

4.7.1. Examples of Comments Definition

4.7.1.1. Example of Short Comment Definition

```
<!-- Short Comments -->
<def:CommentDef OID="COM.DOMAIN.DM">
  <Description>
    <TranslatedText xml:lang="en">See Reviewer's Guide, Section 2.1
      Demographics
    </TranslatedText>
  </Description>
</def:CommentDef>

<def:CommentDef OID="COM.DSDECOD">
  <Description>
    <TranslatedText xml:lang="en">CRF controlled terminology was mapped to match
      CDISC controlled terminology.
    </TranslatedText>
  </Description>
</def:CommentDef>
```

4.7.1.2. Example *of External Comment definition*

```
<ItemGroupDef OID="IG.QSCG" Domain="QS" Name="QSCG" Repeating="Yes" IsReferenceData="No"
   SASDatasetName="QSCG" Purpose="Tabulation"
   def:Structure="One record per questionnaire per question per visit per subject"
   def:Class="FINDINGS"
   def:CommentOID="COM.DOMAIN.QS" def:ArchiveLocationID="LF.QSCG">
…

<ItemGroupDef OID="IG.QSCS" Domain="QS" Name="QSCS" Repeating="Yes" IsReferenceData="No"
  SASDatasetName="QSCS" Purpose="Tabulation"
  def:Structure="One record per questionnaire per question per visit per subject"
  def:Class="FINDINGS"
  def:CommentOID="COM.DOMAIN.QS" def:ArchiveLocationID="LF.QSCS">
…

<!-- Comment Definition: Long Comment, included in a PDF file -->
<def:CommentDef OID="COM.DOMAIN.QS">
  <Description>
    <TranslatedText xml:lang="en"> QS is submitted as a split dataset. The split was done
      based on QSCAT as QSCG (CLINICAL GLOBAL IMPRESSIONS), QSCS (CORNELL SCALE
      FOR DEPRESSION INDEMENTIA) and QSMM (MINI MENTAL STATE EXAMINATION).
      See additional documentation in the Reviewer's Guide, Split Datasets Section.
    </TranslatedText>
  </Description>
  <def:DocumentRef leafID="LF.ReviewersGuide"/>
</def:CommentDef>

<def:leaf ID="LF.ReviewersGuide" xlink:href="reviewersguide.pdf">
  <def:title>Reviewers Guide</def:title>
</def:leaf>
```

5. Specification

5.1. Define-XML Scope

A Define-XML file provides metadata for a set of domains, datasets, variables and associated information described in this document. This can be used for the regulatory submission of SDTM, ADaM or SEND domains for:

- A single clinical study
- A single non-clinical study
- An integrated summary of safety

5.2. Define-XML Structure

Define-XML is an extension to the CDISC ODM standard, and so Define-XML files follow the same basic structure as ODM files.

A Define-XML file includes the following key content components:

- XML header, the ODM root element, Study and MetaDataVersion.
- Information about linked PDF documents such as annotated Case Report Forms and Supplemental Data Definitions.
- Dataset definitions.
- Variable definitions.
- Value definitions (including Where Clause definitions)
- Controlled Terminology definitions
- Computational Method definitions
- Comment definitions

The sections that follow describe what a Define-XML file can contain. Each of the elements is described in the sections below in the order in which they occur in the XML. Elements that may be used in more than one context are presented where they first appear in the document.

Note that the section hierarchy in this document does not reflect the XML structure. For example, the ODM and Study elements are described at the same level in this document, however, the Study element in the XML is a child of the ODM element.

Each section begins with a brief description of the element. This is then followed by an *element table*, and an *attribute table*. In some cases, a section or sub-section concludes with an XML example accompanied by explanation of the example where necessary. However, most examples are provided in Section 4 General Specifications for Define-XML.

An *element table* describes the different aspects of an element's definition while the *attribute table* describes the element's attributes. The following templates illustrate the layouts of these tables, including headers and descriptions of the content.

Element Table Template

Element Name:	*Name of the element*
Element XPath:	*XPath showing where the element belongs in the XML*
Element Textual Value:	*A description of the value of the element. If an element has no text value (e.g. it has child elements instead), then this cell is populated with "None".*
Usage	Requirement: *This is populated with one of three values:* • *"Required" when at least one instance of the element is required* • *"Optional" when the element is optional* • *"Conditional" when at least one instance of the element is required under certain conditions. It will include the conditions under which the element is Required.* • **Refer to Section 3.6 Required Components for a description of XML requirement's values.** Cardinality: *This indicates the number of instances expected (e.g.* "Exactly One", "One or more", *etc.)* Business Rule(s): *This is populated with rules that have to be satisfied in addition to an XML schema validation for a define.xml document to be considered compliant with the Define-XML v2.0.0 specification.* Other Information: *This is populated with any other information about the element, including the conditions under which the element is included, how the schema is applied to support the model, relative position of the element in the model, etc*.
Attributes:	*A comma-delimited list of the attributes of this element. If the element has no attributes, this is populated with* "None".
Child Elements:	*A comma-delimited list of the immediate child elements of this element. If the element has no child elements, this is populated with "None". The order of child elements shown in the specification is the order in which they must appear in a define.xml file.* *A link to a child element will be provided when the child element is described in a different section of the document and not under a sub-section of the element being described or in the section or subsection immediately following the current element.*

Attribute Table Template

Attribute	Usage	Allowable Values	Description
Name of the attribute	*This is populated with* "Required" *when the attribute is required, "Optional" when the attribute is optional, or* "Conditional" *when the attribute is required under certain conditions.* *It will include the conditions under which the attribute is Required.* *Default: This will be populated with a default value if one is provided in the specification.*	*Any combination of the following:* Allowable Value: *The only allowed value* Allowable Values: *A comma-delimited list of the allowable values* Value Description: *A textual description of allowable values* See Appendix *xx: A reference to an appendix including a hyperlink to the appendix* Sample: *An example*	*A textual description of the attribute beyond what is included in the Allowable Values column.* Business Rule(s): *Rules that have to be satisfied in addition to schema validation for a define.xml document to be considered compliant with the Define-XML v2.0.0 specification.*

The ODM elements AdminData, ClinicalData, ReferenceData, and Associations are not valid in Define-XML. The ODM ds:Signature element should not be used when Define-XML is used in a regulatory submission context.

5.3. Define-XML Specification Details

5.3.1. XML Header

All XML files must begin with an XML header, so the first line of a *define.xml* file must be an XML header. The XML header indicates to applications that the remainder of the file is XML and specifies character encoding it uses.

5.3.1.1. Example XML Header

```
<?xml version="1.0" encoding="UTF-8"?>
```

This example shows a *define.xml* using the "UTF-8" character encoding.

5.3.2. Stylesheet Reference

An XSL stylesheet can optionally be referenced between the XML header and the ODM element. This allows the *define.xml* file to be easily viewed in a web browser. If a stylesheet reference is provided then a browser can open the *define.xml* file and display it according to the stylesheet. For a browser to correctly show the *define.xml* the referenced stylesheet must exist at the location specified. If a relative location is given for the stylesheet, it is relative to the location of the *define.xml* file.

The Define-XML standard does not dictate how a stylesheet should display a *define.xml* file. An example stylesheet is provided, however this can be altered to satisfy alternate visualization needs.

If a stylesheet reference is not provided, a browser will display the XML contents of the *define.xml* file.

5.3.2.1. Example Stylesheet Reference

```
<?xml-stylesheet type="text/xsl" href="define2-0-0.xsl"?>
```

This example references a stylesheet contained in the same location as the *define.xml* file.

5.3.3. ODM Element

The first XML element in a file is known as the root element. For Define-XML files, the root element is the ODM element.

This element identifies the namespaces used and includes attributes that affect the processing of the document as a whole.

Element Name:	ODM
Element XPath:	/ODM
Element Textual Value:	*None*
Usage	Requirement: Required Cardinality: Exactly one Other Information: This is the root element for the *define.xml* document
Attributes:	xmlns, xmlns:def, xmlns:xlink, xmlns:xsi, xsi:schemalocation, ODMVersion, FileType, FileOID, CreationDateTime, AsOfDateTime, Originator, SourceSystem, SourceSystemVersion
Child Elements:	Study

Attribute	Usage	Allowable Values	Description
xmlns	Required	"http://www.cdisc.org/ns/odm/v1.3"	Identifies the default namespace for this document.
xmlns:def	Required	"http://www.cdisc.org/ns/def/v2.0"	XML namespace for Define-XML v2.0.0. While "def:" is suggested prefix for the Define-XML namespace, it should not be relied upon by the receiving application.
xmlns:xlink	Conditional Required when xlink:href is provided.	"http://www.w3.org/1999/xlink"	XML namespace for XLink.
xmlns:xsi	Conditional Required when xsi:schemalocation is provided.	"http://www.w3.org/2001/XMLSchema-instance"	XML Schema instance namespace. Required when xsi:schemalocation is provided.

xsi:schemalocation	Optional	Text Sample: "http://www.cdisc.org/ns/def/v2.0 define2-0-0.xsd"	Identifies the location of the schema for this XML document. First part is the Namespace URI the second part is the location of the schema either on the internet (e.g. http://www.abc.com/def.xsd) or on the local file system (e.g. def.xsd). Using a local copy of the schema rather than referencing a schema using a URL on the web is recommended as it improves the probability that the software validating the Define-XML instance can find and access the appropriate files. However, when submitting a Define-XML to a regulatory authority, be aware that relative file references or references to a shared drive on a local area network may not work when the submission contents are transmitted to a different network location.
ODMVersion	Required	"1.3.2"	Identifies the ODM version that underlies the schema for the Define-XML document. ODMVersion is optional in the ODM standard, but required in Define-XML.
FileType	Required	"Snapshot"	Define-XML documents do not include audit trail elements, so the FileType is Snapshot .
FileOID	Required	Text	A unique identifier for this file. See the ODM specification for a discussion of FileOID recommendations.
CreationDateTime	Required	ISO8601 datetime Sample: "2010-09-30T15:31:04"	The date and time when the specific version of the *define.xml* file was created. This can be more accurately thought of as the "last modified" date and time.
AsOfDateTime	Optional	ISO8601 datetime Sample: "2010-09-30T15:31:04"	The date and time at which the source database was queried to create this document.
Originator	Optional	Text Sample: "Company XYZ"	Submission sponsor name.
SourceSystem	Optional	Text	The name of the application that generated the *define.xml* file.
SourceSystem Version	Optional	Text	The version of the "SourceSystem" above.

5.3.3.1. Example XML Header, Stylesheet Reference and ODM Element

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="../Stylesheets/define2.xsl"?>
<ODM
  xmlns="http://www.cdisc.org/ns/odm/v1.3"
  xmlns:def="http://www.cdisc.org/ns/def/v2.0"
  xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.cdisc.org/ns/def/v2.0
                      ../schema/cdisc-define-2.0/define2-0-0.xsd"
  ODMVersion="1.3.1"
  FileType="Snapshot"
  FileOID="Studycdisc01-Define2-XML_2.0.0"
  CreationDateTime="2012-07-21T11:27:02"
  Originator="CDISC XML Technologies Team">
```

5.3.4. Study Element

Study is the first element in the Define-XML document after the ODM element.

Element Name:	Study
Element XPath:	/ODM/Study
Element Textual Value:	*None*
Usage	Requirement: Required Cardinality: Exactly One Other Information: The child element GlobalVariables contains child elements that capture high level study information. The child element MetaDataVersion includes child elements to describe a collection of Datasets.
Attributes:	StudyOID
Child Elements:	GlobalVariables, MetaDataVersion

Attribute	Usage	Allowable Values	Description
StudyOID	Required	Text	The unique ID of the Study. See the ODM specification section 2.11 for OID considerations.

5.3.4.1. GlobalVariables Element

GlobalVariables is the first child of Study.

Element Name:	GlobalVariables
Element XPath:	/ODM/Study/GlobalVariables
Element Textual Value:	*None*
Usage	Requirement: Required Cardinality: Exactly One Other Information: High level study information.
Attributes:	None
Child Elements:	StudyName, StudyDescription, ProtocolName

5.3.4.2. StudyName Element

StudyName is the first child of GlobalVariables.

Element Name:	StudyName
Element XPath:	/ODM/Study/GlobalVariables/StudyName
Element Textual Value:	*The short, external name assigned to the Study*
Usage	Requirement: Required Cardinality: Exactly One
Attributes:	None
Child Elements:	None

5.3.4.3. StudyDescription Element

StudyDescription is the second child of GlobalVariables.

Element Name:	StudyDescription
Element XPath:	/ODM/Study/GlobalVariables/StudyDescription
Element Textual Value:	*A text description of the contents of the Study.*
Usage	Requirement: Required Cardinality: Exactly One Other Information: Usually found in the high level description of the study the Protocol document.
Attributes:	None
Child Elements:	None

5.3.4.4. ProtocolName Element

ProtocolName is the third child of GlobalVariables.

Element Name:	ProtocolName
Element XPath:	/ODM/Study/GlobalVariables/ProtocolName
Element Textual Value:	*The sponsor's internal name assigned to the Study*
Usage	Requirement: Required Cardinality: Exactly One Other Information: Usually found in the high level description of the study in the Protocol document.
Attributes:	None
Child Elements:	None

5.3.4.5. Example Study and GlobalVariables Elements

```
<Study OID="cdisc01">
  <GlobalVariables>
    <StudyName>CDISC01</StudyName>
    <StudyDescription>CDISC Test Study</StudyDescription>
    <ProtocolName>CDISC01</ProtocolName>
  </GlobalVariables>
```

5.3.5. MetaDataVersion Element

The MetaDataVersion element contains all the definitions related to the Domains specified in the *define.xml*. It includes attributes to identify the versions of the CDISC SDTM and Define-XML standards used by the submission and can also contain links to external documents that are frequently included – such as the Annotated Case Report form and the Supplemental Data Definition document.

The table below specifies how the MetadataVersion element in a Define-XML file shall be constructed.

Element Name:	MetaDataVersion
Element XPath:	/ODM/Study/MetaDataVersion
Element Textual Value:	*None*
Usage	Requirement: Required Cardinality: One
Attributes:	OID, Name, Description, def:DefineVersion, def:StandardName, def:StandardVersion
Child Elements:	def:AnnotatedCRF, def:SupplementalDoc, def:ValueListDef, def:WhereClauseDef, ItemGroupDef, ItemDef, CodeList, MethodDef, def:CommentDef, def:leaf

Attribute	Usage	Allowable Values	Description
OID	Required	Text	Unique ID for the MetaDataVersion. See the ODM specification section 2.11 for OID considerations.
Name	Required	Text	Name for the MetaDataVersion.
Description	Optional	Text Sample: "Study CDISC01 Data Definitions"	Description for the MetaDataVersion.
def:DefineVersion	Required	"2.0.0"	Version of Define-XML that the file conforms to.
def:StandardName	Required	Text Allowable Values (Extensible): SDTM-IG ADaM-IG SEND-IG	Name of an external standard to which the data conforms.

def:StandardVersion	Required	Text	Version of an external standard to which the data conforms. See Section 1.5 for a list of the versions of CDISC Standards supported by Define-XML v2.0.0 (e.g. "3.1.2" for SDTM-IG).

5.3.5.1. Example MetaDataVersion Element

The XML in this example indicates that this *define.xml* instance follows Version 2.0.0 of the Define-XML specification and Version 3.1.2 of the CDISC Study Data Tabulation Model (SDTM) standard.

```
<MetaDataVersion OID="CDISC01.SDTMIG.3.1.2.SDTM.1.2"
  Name="Study CDISC01, Data Definitions"
  Description="Study CDISC01, Data Definitions"
  def:DefineVersion="2.0.0"
  def:StandardName="SDTM-IG" def:StandardVersion="3.1.2">
```

5.3.6. def:AnnotatedCRF Element

An Annotated Case Report Form (CRF) is a Portable File Format (PDF) document that provides the mapping of data collection fields to the variables or discrete variable values contained within the datasets.

Element Name:	def:AnnotatedCRF
Element XPath:	/ODM/Study/MetaDataVersion/def:AnnotatedCRF
Element Textual Value:	*None*
Usage	Requirement: Optional Cardinality: Zero or One Other Information: Contains the DocumentRef for the Annotated CRF.
Attributes:	None
Child Elements:	def:DocumentRef

5.3.6.1. def:DocumentRef Element

The DocumentRef element is included in the def:AnnotatedCRF, def:SupplementalDoc, def:Origin, MethodDef and def:CommentDef elements. The document Ref references the def:leaf element that identifies the PDF document file.

Element Name:	def:DocumentRef
Element XPath:	/ODM/Study/MetaDataVersion/def:AnnotatedCRF/def:DocumentRef /ODM/Study/MetaDataVersion/def:SupplementalDoc/def:DocumentRef /ODM/Study/MetaDataVersion/ItemDef/def:Origin/def:DocumentRef /ODM/Study/MetaDataVersion/MethodDef/def:DocumentRef /ODM/Study/MetaDataVersion/def:CommentDef/def:DocumentRef
Element Textual Value:	*None*

Usage	Requirement: Conditional Required for def:AnnotatedCRF and def:SupplementalDoc Optional for def:Origin, MethodDef and def:CommentDef Cardinality: One or More Business Rule: If multiple documents are provided there will be multiple def:DocumentRef child elements within the def:SupplementalDoc element.
Attributes:	leafID
Child Elements:	def:PDFPageRef

Attribute	Usage	Allowable Values	Description
leafID	Required	Text	Reference to the unique ID of the def:leaf element that contains the location of a PDF file.

5.3.6.1.1. def:PDFPageRef Element

This element is the container for PDF page references.

Element Name:	def:PDFPageRef
Element XPath:	/ODM/Study/MetaDataVersion/ItemDef/def:Origin/def:DocumentRef/def:PDFPageRef /ODM/Study/MetaDataVersion/MethodDef/def:DocumentRef/def:PDFPageRef /ODM/Study/MetaDataVersion/def:CommentDef/def:DocumentRef/def:PDFPageRef
Element Textual Value:	*None*
Usage	Requirement: Conditional • Required for def:Origin/@Type="CRF" • Optional in all other cases Cardinality: Zero or more.
Attributes:	Type, PageRefs, FirstPage, LastPage
Child Elements :	*None*

Attribute	Usage	Allowable Values	Description
Type	Required	Allowable Values: PhysicalRef NamedDestination	Type of page for page references indicated in the PageRefs attribute. Business Rule: When Type="NamedDestination", NamedDestinations have to be created within the PDF document to be able to link to them with a hyperlink.
PageRefs	Optional	Text Sample: "17 20 32"	List of PDF pages separated by a space.
FirstPage	Conditional Required if PageRefs is not provided	Integer	First page in a range of pages. Note that the way to indicate the range of pages depends on the associated Type attribute provided.
LastPage	Conditional Required if PageRefs is not provided	Integer	Last page in a range of pages. Note that the way to indicate the range of pages depends on the associated Type attribute provided.

5.3.7. def:SupplementalDoc Element

Contains the DocumentRef for each Supplemental Document.

Element Name:	def:SupplementalDoc
Element XPath:	/ODM/Study/MetaDataVersion/def:SupplementalDoc
Element Textual Value:	*None*
Usage	Requirement: Optional Cardinality: Zero or One Business Rule: If multiple documents are provided there will be multiple def:DocumentRef child elements within the def:SupplementalDoc element.
Attributes:	None
Child Elements:	def:DocumentRef

5.3.8. def:ValueListDef Element

The table below specifies the XML structure for Value List metadata. This construct is an extended element within the Define-XML schemas.
Value List defintion.

Element Name:	def:ValueListDef
Element XPath:	/ODM/Study/MetaDataVersion/def:ValueListDef
Element Textual Value:	*None*
Usage	Requirement: Conditional Cardinality: • Required for each unique value of the ValueListOID attribute within the MetaDataVersion. Business Rule: • A def:ValueListDef element to describe the variable QNAM must be included for each ItemGroupDef element that has def:Class="RELATIONSHIP". Other Information: it is the container for child ItemRef elements that link to ItemDef elements for a complete definition of a Value-Level Variable.
Attributes:	OID
Child Elements :	*ItemRef*

Attribute	Usage	Allowable Values	Description
Attributes			
OID	Required	Text	Unique ID for the Value List. See the ODM specification section 2.11 for OID considerations.

5.3.8.1. ItemRef Element

For each Variable within a dataset, the ItemGroupDef must include an ItemRef element. ItemRef elements are also included in ValueListDef elements.

Element Name:	Description
Element XPath(s):	/ODM/Study/MetaDataVersion/ItemGroupDef/ItemRef /ODM/Study/MetaDataVersion/def:ValueListDef/ItemRef
Element Textual Value:	*None*
Usage	Requirement: Required Cardinality: • One for each dataset variable (when parent node is an ItemGroupDef element) • One for each Value to be defined (when parent node is a ValueListDef element).
Attributes:	ItemOID, OrderNumber, Mandatory, KeySequence, Role, RoleCodeListOID, MethodOID
Child Elements :	def:WhereClauseRef (valid only when parent node is a ValueListDef).

Attribute	**Usage**	**Allowable Values**	**Description**
ItemOID	Required	Text	Reference to the unique ID of an ItemDef element. See the ODM specification section 2.11 for OID considerations. Business Rule: • Each ItemOID used in an ItemRef must match an OID for an ItemDef.

OrderNumber	Optional	Integer	Display order of the variable within the domain dataset. • Business Rules:See section 4.1.1.4 of the SDTM-IG for variable ordering requirements for the SDTM or section 3 of the ADaM Implementation Guide for information about variable ordering for ADaM datasets. If OrderNumber is not provided, displays will list Items in the order the ItemRef elements appear within the ItemGroup or ValueList. If the OrderNumber is provided, displays should list Items in the order given by the OrderNumber within the ItemGroup or ValueList.

Mandatory	Required	Allowable Values: Yes No	Business Rules: • For SDTM based variables where Core is "Req" (Required), the value of Mandatory should be set to "Yes". • For variables where Core is "Exp" (Expected) or "perm" (Permissible) and the sponsor does not require a more restrictive condition, Mandatory should be set to "No". • Variables where Mandatory="Yes" must not have a null value.
KeySequence	Conditional Required for Regulatory Submissions	Integer	The KeySequence indicates that this item is a key for the enclosing item group. It also provides an ordering for the keys.
MethodOID	Conditional This attribute and the associated MethodDef are Required when the Type attribute for the def:Origin child element of the referenced ItemDef is 'Derived'. Otherwise, this attribute is Optional.	Text	Reference to the unique ID of a MethodDef element.

Role	Optional for SDTM standard domains. The values provided by the SDTM or SEND IGs are used for the standard domains. Conditional required for SDTM custom domains Not applicable for ADaM.	Text Allowable Values: If SDTM or SEND datasets, any valid SDTM or SEND role as defined in the corresponding IG. For other datasets any value can be used.	Variable Role defines how the variable defined by the corresponding ItemDef element is used within the dataset. Note that Role is not defined for ADaM variables or parameters at the present time.
RoleCodeListOID	Optional	Text	• Reference to the unique ID of a CodeList element that defines the values for the codes given in the Role attribute.

5.3.8.2. def:WhereClauseRef Element

The def:WhereClauseRef is a new element in version 2.0.0. It references a WhereClauseDef element.

Element references the def:WhereClauseDef that describes the conditions under which the Variable Values are defined by the referenced ItemDef.

If more than one is provided, the parent def:ValueList Def applies for multiple **Slices**.

Element Name:	def:WhereClauseRef
Element XPath:	/ODM/Study/MetaDataVersion/def:ValueListDef /ItemRef/def:WhereClauseRef
Element Textual Value:	*None*
Usage	Requirement: Conditional Cardinality: • One or more def:WhereClauseRef elements is Required for each ItemRef child element within a def:ValueListDef Business Rule: • Not allowed as a child element of an ItemRef element if the parent node is a def:ItemGroupDef element. It will be considered non-conforming
Attributes:	WhereClauseOID
Child Elements :	*None*

Attribute	Usage	Allowable Values	Description
WhereClauseOID	Required	Text	• Reference to the unique ID of a def:WhereClauseDef element.

5.3.9. def:WhereClauseDef Element

The table below specifies the XML structure for a Where Clause. This construct is an extended element within the Define-XML schemas.
Where Clause Definition.
One for each def:WhereClauseRef included in an ItemRef in this MetaDataVersion.

Element Name:	def:WhereClauseDef
Element XPath:	/ODM/Study/MetaDataVersion/def:WhereClauseDef
Element Textual Value:	*None*
Usage	Requirement: Conditional Cardinality: • A def:WhereClause is required for each unique value of the WhereClauseOID attribute value in a def:WhereClauseRef element within the MetaDataVersion.
Attributes:	OID, def:CommentOID
Child Elements :	*RangeCheck*

Attribute	Usage	Allowable Values	Description
OID	Required	Text	Unique ID for the WhereClauseDef. See the ODM specification section 2.11 for OID considerations.
def:CommentOID	Conditional Required when RangeCheck includes def:ItemOID values that belong to different ItemGroupDef elements	Text	Reference to the unique ID of a def:CommentDef that describes how to join the datasets when the WhereClause includes references to variables in different datasetst.

5.3.9.1. RangeCheck Element

Element Name:	RangeCheck
Element XPath:	/ODM/Study/MetaDataVersion/def:WhereClauseDef/RangeCheck
Element Textual Value:	Contains the comparison specification defining the Where Clause condition.
Usage	Requirement: Required Cardinality: • Each def:WhereClauseDef element must have at least one RangeCheck child element. Other Information: • If multiple RangeChecks are given the condition is the logical AND of all the RangeChecks.
Attributes:	Comparator, SoftHard, def:ItemOID
Child Elements :	*CheckValue*

Attribute	Usage	Allowable Values	Description
Comparator	Required	Allowable Values: LT LE GT GE EQ NE IN NOTIN	Comparison operator for Where Clause.
SoftHard	Required	Allowable Values: Soft Hard	If an actual data value fails the constraint, it is either rejected (a Hard constraint) or a warning is produced (a Soft constraint). Business Rule: The SoftHard attribute has no meaning in the Define-XML context. Although ODM requires a value equal to "Hard" or "Soft", neither value implies any meaning to the enclosing RangeCheck or WhereClauseDef element.
def:ItemOID	Required	Text	Reference to the unique ID of an ItemDef that is used to compare with the CheckValue.

5.3.9.2. CheckValue Element

Element Name:	RangeCheck
Element XPath:	/ODM/Study/MetaDataVersion/def:WhereClauseDef/RangeCheck
Element Textual Value:	The comparison value
Usage	Requirement: Required Cardinality: One
Attributes:	*None*
Child Elements :	*None*

5.3.10. ItemGroupDef Element

The ItemGroupDef element with the set of attributes and child elements as described below is used to describe the dataset metadata in XML.

One for each dataset included within the study submitted.

Element Name:	ItemGroupDef
Element XPath:	/ODM/Study/MetaDataVersion/ItemGroupDef
Element Textual Value:	*None*
Usage	Requirement: Required Cardinality: One or more Business Rule: • The Implementation Guides for each relevant standard specify the datasets required for FDA submissions.
Attributes:	OID, Name, Repeating, IsReferenceData, SASDatasetName, Domain, Purpose, def:Structure, def:Class, def:ArchiveLocationID, def:CommentOID
Child Elements:	Description, ItemRef, Alias, def:leaf

Attribute	Usage	Allowable Values	Description
OID	Required	Text	Unique ID for the ItemGroupDef (dataset). See the ODM specification section 2.11 for OID considerations.
Name	Required	Text	Short description for the ItemGroup. Note that the Name attribute must be the same as SASDatasetName if SAS is being used as a transport mechanism. If the transport mechanism is not SAS based this attribute will contain the name of the domain.
Domain	Conditional • Required in the context of a regulatory submission in Define-XML for SDTM and SEND • Not applicable for ADaM	Text	Domain as specified by the SDTM Metadata Submission Guidelines. For split domains, Domain must contain the root Domain name.

SASDatasetName	Conditional • Required in the context of a regulatory submission in Define-XML	Text	Root name of SAS dataset contained in the SAS Transport file containing the ItemGroup data. This is the same as the file name but without the ".xpt" extension. Business Rules: • Must conform to SAS Transport file naming rules. • If a value is provided it should be the same as the Domain name specified in the Name attribute above, except for split datasets. • In the case of split datasets, the value of the SASDatasetName attribute and the file name referenced by the def:ArchiveLocationID attribute will contain a suffix component to identify the individual split file; i.e, QSCG, QSCS, etc.
Repeating	Required	Allowable Values: Yes No	• Indicates whether a domain contains more than one record per subject or only one record per subject. Business Rules: • Set Repeating='No' when used in the reference data section
IsReferenceData	Optional Default Value: No	Allowable Values: Yes No Sample: For Trial Design Domains: Yes For Demographics Domain: No	Indicates whether the dataset contains reference data (not subject data)

Purpose	Required	Allowable values: Tabulation, Analysis.	Purpose of domain or dataset. Business Rules: • For SDTM and SEND use Tabulation • For ADaM use Analysis
def:Structure	Required	Text Samples: MH domain: "One record per medical history event per subject" VS domain: "One record per vital sign measurement per visit per subject" ADQSADAS dataset: "One record per subject per parameter per analysis visit per analysis date"	Description of the level of detail represented by individual records in the dataset.
Def:Class	Conditional • Required in the context of a regulatory submission in Define-XML	Text Allowable Values: CDISC Submission Value (=CodedValue) as listed in the NCI/CDISC Controlled Terminology, CodeList GNRLOBSC – General Observation Class, NCI Code C103329 SDTM and SEND: SPECIAL PURPOSE FINDINGS EVENTS INTERVENTIONS TRIAL DESIGN RELATIONSHIP ADaM: SUBJECT LEVEL ANALYSIS DATASET BASIC DATA STRUCTURE ADAM OTHER	General class of the data domain. Notes: - The allowable values are case sensitive and must be used as specified. - The CodeList is not extensible.

def:ArchiveLocationID	Conditional • Required in the context of a regulatory submission in Define-XML	Text	Reference to the unique ID of a def:leaf that provides the actual location and file name of the SAS transport file. If provided, it should match the leaf:id attribute of the def:leaf child element. If not provided, the root dataset file name is expected to be the same as the Name attribute.
def:CommentOID	Optional	Text	Reference to the unique ID of a def:CommentDef element that contains the comment for the ItemGroupDef (dataset). For ADaM datasets, the comment referenced by this value is the dataset Documentation metadata. Refer to the Comments section (Section 4.7 of this document) **Note** the specification change. The Comment attribute, allowed in ODM, is not valid in Define-XML v2.0.0. It has been replaced by the reference to the definition of the comment.

5.3.10.1. Description Element

Description is an ODM version 1.3 element and is a new component of Define-XML in Version 2.0.0. It is used to provide a short text description of the element contents.

Note the specification change. It replaces the def:Label requirement for ItemDef and ItemGroupDef in the V1 specification.

Element Name:	Description
Element XPath(s):	/ODM/Study/MetaDataVersion/ItemGroupDef/Description /ODM/Study/MetaDataVersion/ItemDef/Description /ODM/Study/MetaDataVersion/ItemDef/def:Origin/Description /ODM/Study/MetaDataVersion/MethodDef/Description /ODM/Study/MetaDataVersion/def:CommentDef/Description
Element Textual Value:	*None*
Usage	Requirement: Conditional • Required for regulatory submissions or any case where the Define-XML will be checked for compliance with a standard that requires a dataset label. • Required for MethodDef/Description and def:CommentDef/Description • Optional for ItemDef/Description corresponding to Value Level definitions or for def:Origin/Description. Cardinality: Zero or One Business Rules: • For SDTM or SEND standard domains or for ADaM standard datasets, the ItemGroupDef/Description and ItemDef/Description should exactly match the label for the relevant standard dataset or variable. • For custom domains or datasets, sponsors should provide a short description of the type of data contained within the dataset or variable in ItemGroupDef/Description and ItemDef/Description. • def:Origin/Description must be provided to indicate the source when def:Origin@Type="Predecessor". Conventions described in IG documents for the specific standard should be followed.
Attributes:	*None*
Child Elements :	TranslatedText

5.3.10.1.1. TranslatedText Element

Element Name:	TranslatedText
Element XPath(s):	/ODM/Study/MetaDataVersion/ItemGroupDef/Description/TranslatedText /ODM/Study/MetaDataVersion/ItemDef/Description/TranslatedText /ODM/Study/MetaDataVersion/CodeList/CodeListItem/Decode/TranslatedText /ODM/Study/MetaDataVersion/MethodDef/Description/TranslatedText /ODM/Study/MetaDataVersion/def:CommentDef/Description/TranslatedText
Element Textual Value:	*text string*

Usage	Requirement: Required Cardinality: One or more. • Multiple TranslatedText child elements can be used to provide the dataset description in different languages. One for each language the description is desired. Business Rules: • A child TranslatedText element in English (without attribute xml:lang or xml:lang="en") is required when files are submitted to the FDA. • In cases where SAS Transport files are the transport format, the value provided in the ItemGroupDef element should match the dataset label provided in the SAS Transport file.
Attributes:	xml:lang
Child Elements :	*None*

Attribute	Usage	Allowable Values	Description
xml:lang	Optional Default: "en"	Allowable Values: see: http://www.rfc-editor.org/rfc/bcp/bcp47.txt Samples: "en" for English "en-GB" for British English	Code representing the language of the enclosed text value. Business Rule: xml:lang should be unique within parent element.

5.3.10.2. Alias Element

The Alias element provides a reference to the Domain description in the case where an ItemGroupDef represents a split domain or a reference to a C-Code when a CodeList, CodeListItem or EnumeratedItem corresponds to a CDISC Controlled Terminology.

Element Name:	Alias
Element XPath(s):	/ODM/Study/MetaDataVersion/ItemGroupDef/Alias /ODM/Study/MetaDataVersion/CodeList/Alias /ODM/Study/MetaDataVersion/CodeList/CodeListItem/Alias /ODM/Study/MetaDataVersion/CodeList/EnumeratedItem/Alias
Element Textual Value:	*None*
Usage	Requirement: Conditional Cardinality: One or more • When used for regulatory submission, the Alias element is required for each ItemGroup that is part of a Domain that has been split into different datasets. • When used for regulatory submission of SDTM, ADaM or SEND metadata, the Alias element is required for each CodeList that represents a CDISC Controlled Terminology and for each EnumeratedITem of CodeListItem element that represents a CDISC defined term in a CDISC Controlled Terminology.
Attributes:	Context, Name
Child Elements :	*None*

Attribute	Usage	Allowable Values	Description
Context	Required	Text Allowable Values: • As a child element of an ItemGroupDef element: DomainDescription • As a child element of a CodeList, CodeListItem or EnumeratedItem element: nci:ExtCodeID	Indicates the context or setting where the Alias Name attribute applies.
Name	Required	Text Allowable Values: • As a child element of an ItemGroupDef element representing a split domain, when Context="DomainDescription": Description of the parent domain. • As a child element of a CodeList element, when Context="nci:ExtCodeID": C-Code for corresponding CDISC Controlled Terminology codelist. • As a child element of a CodeListItem or EnumeratedItem, when Context="nci:ExtCodeID": C-Code for corresponding CDISC Controlled Terminology Term.	Alternative Name for parent element.

5.3.11. ItemDef Element

For each Variable and Value definition an ItemDef element should be provided. Use one ItemDef element per unique variable.

Element Name:	ItemDef
Element XPath:	/ODM/Study/MetaDataVersion/ItemDef
Element Textual Value:	*None*
Usage	Requirement: Required Cardinality: • An ItemDef element is required for each ItemOID value that appears in an ItemRef contained in a MetaDataVersion..
Attributes:	OID, Name, DataType, Length, SignificantDigits, SASFieldName, def:DisplayFormat, def:CommentOID
Child Elements :	*Description, CodeListRef, def:Origin, def:ValueListRef*

Attribute	Usage	Allowable Values	Description
OID	Required	Text	Unique ID for the ItemDef (variable/value). See the ODM specification section 2.11 for OID considerations.
Name	Required	Text	Dataset Variable name or Variable Value Name.
DataType	Required	Allowable Values: Refer to Section 4.2.1 for a list of the valid Define-XML DataType values. Samples: For SDTM, SEND and ADaM variables one of ("text", "float", "integer", "date", "datetime").	The data type of the variable or value.
Length	Conditional Required if DataType is "text", "integer", or "float".	Integer	The variable length. Business Rule: Length should be defined as the maximum expected variable length. Should only be present for DataType equal to "text", "integer", or "float".

SignificantDigits	Conditional Required if DataType is "float".	Integer	The number of digits following the decimal point in a floating point number. Business Rule: When DataType is float both Length and SignificantDigits must be provided.
SASFieldName	Conditional Required in the context of a regulatory submission in Define-XML	Text	SAS Variable Name. Business Rule: Follow rules for Variable names in SAS Transport files.
def:DisplayFormat	Optional	Text Samples: 8.2 (means display floating point variable values to the second decimal place). date9. (SAS date9 format means display a numeric ADaM date variable in the format DDMMMYYYY).	Display format supports data visualization of numeric float and date values.

def:CommentOID	Optional	Text	Reference to the unique ID of a def:CommentDef element that contains the comment for the ItemDef (variable). Refer to the Comments section (Section 4.7 of this document) **Note** the specification change. The Comment attribute, allowed in ODM, is no longer valid in Define-XML v2.0.0. It has been replaced by the reference to the definition of the comment. Business Rule: Must match the OID of a def:CommentDef element in the same MetaDataVersion.

5.3.11.1. CodeListRef Element

Link to CodeList defining the variable controlled terminology.

Element Name:	CodeListRef
Element XPath:	/ODM/Study/MetaDataVersion/ItemDef/CodeListRef
Element Textual Value:	*None*
Usage	Requirement: Optional Cardinality: One Business Rule: • If a variable or value definition includes Controlled Terminology a CodeList element should be provided as a child element on the ItemDef.
Attributes:	CodeListOID
Child Elements :	*None*

Attribute	Usage	Allowable Values	Description
CodeListOID	Required	Text	Reference to the unique ID of a CodeList element that defines Controlled Terminology for the variable or values defined by the ItemDef. See the ODM specification section 2.11 for OID considerations. Business Rule: Must match the OID of a CodeList element in the same MetaDataVersion.

5.3.11.2. def:ValueListRef Element

It is the OID of the def:ValueListDef that contains the Value List definition associated with the variable.
If Value Level Metadata is required for a Variable, a def:ValueListRef element should be provided as a child element on the ItemDef for the variable definition.

Element Name:	def:ValueListRef
Element XPath:	/ODM/Study/MetaDataVersion/ItemDef/def:ValueListRef
Element Textual Value:	*None*
Usage	Requirement: Optional Cardinality: One
Attributes:	ValueListOID
Child Elements :	*None*

Attribute	Usage	Allowable Values	Description
ValueListOID	Required	Text	Reference to the unique ID of a def:ValueListDef element that provides value level metadata. See the ODM specification section 2.11 for OID considerations. Business Rule: Must match the OID of a def:ValueListDef in the same MetaDataVersion.

5.3.11.3. def:Origin Element

The def:Origin element is new in Define-XML 2.0.0. It is intended to be used in place of the ODM Origin attribute. That is, it is intended to define the Origin metadata for variables in SDTM or SEND datasets or Source metadata for ADaM datasets. The use of the v1.0 ODM Origin attribute within Define-XML files has been deprecated, and is not valid in Define-XML v2.0.0

Element Name:	def:Origin
Element XPath:	/ODM/Study/MetaDataVersion/ItemDef/def:Origin
Element Textual Value:	*None*
Usage	Requirement: Conditional • For regulatory submissions, def:Origin metadata must be provided for all SDTM, ADaM or SEND variables. It is at the sponsor's discretion whether to provide def:Origin at the Variable or Value level. Cardinality: Zero or One. Business Rule: • If the ItemDef corresponding to a SDTM, ADaM or SEND variable includes a def:ValueListRef and all of the ItemDef elements referenced in the corresponding def:ValueListDef include a def:Origin element, the def:Origin is optional with the variable level ItemDef. • If the ItemDef corresponding to a SDTM, ADaM or SEND variable includes a def:ValueListRef and the def:Origin elements of ItemDef elements referenced in the corresponding def:ValueListDef are different, then the def:Origin can not be provided with the variable level ItemDef. • If the variable or value is derived, the corresponding ItemDef must include a MethodOID attribute that references the corresponding MethodDef. • When def:Origin/@Type="CRF", there must be a def:DocumentRef child element and def:DocumentRef/@leafID must match the ID attribute of the def:leaf element corresponding to the def:AnnotatedCRF within the same MetaDataVersion. Otherwise, def:DocumentRef/@leafID must match the ID of a defined def:leaf element within the same MetaDataVersion.
Attributes:	Type
Child Elements :	def:DocumentRef, Description

Attribute	Usage	Allowable Values	Description
Type	Required	Value Description: Origin or data source type. Allowable Values: **CRF**: Data that was collected as part of a CRF and has an annotated CRF associated with the variable. **Derived**: Data that is not directly collected on the CRF or received via eDT, but is calculated by an algorithm or reproducible rule defined by the sponsor, which is dependent upon other data values. **Assigned**: Data that is determined by individual judgment (by an evaluator other than the subject or investigator), rather than collected as part of the CRF, eDT or derived based on an algorithm. This may include third party attributions by an adjudicator. Coded terms that are supplied as part of a coding process (as in --DECOD) are considered to have an Origin of "Assigned". Values that are set independently of any subject-related data values in order to complete SDTM fields such as DOMAIN and --TESTCD are considered to have an Origin of "Assigned". **Protocol**: Data that is defined as part of the Trial Design preparation. An example would be VSPOS (Vital Signs Position), which may be specified only in the protocol and not appear on a CRF or transferred via eDT. **eDT**: Data that is received via an electronic Data Transfer (eDT) and usually does not have associated annotations. An origin of eDT refers to data collected via data streams such as laboratory, ECG, or IVRS. **Predecessor**: Data that is copied from a variable in another dataset. For example, predecessor is used to link ADaM data back to SDTM variables to establish traceability.	Business Rule: If the variable is derived, a MethodDef must be provided. The list of allowable values is not extensible.

5.3.12. CodeList Element

For each Controlled Terminology referenced by variable or ValueList, a CodeList element with the definition of the Controlled terminology must be provided. Use one CodeList element per controlled terminology.

Element Name:	CodeList
Element XPath:	/ODM/Study/MetaDataVersion/CodeList
Element Textual Value:	*None*
Usage	Requirement: Conditional Cardinality: A CodeList element must be provided for each distinct value of the CodelistOID attribute in a CodeListRef element in the MetaDataVersion.
Attributes:	OID, Name, DataType, SASFormatName
Child Elements :	EnumeratedItem, CodeListItem, ExternalCodeList, Alias

Attribute	Usage	Allowable Values	Description
OID	Required	Text	Unique ID for the CodeList. See the ODM specification section 2.11 for OID considerations.
Name	Required	Text	Controlled Terminology name. Business Rule: For NCI/CDISC Controlled Terminology, this must exactly match the CodeList Name from the published Controlled Terminology ODM.
DataType	Required	Allowable Values: text float integer	The data type of the codes.
SASFormatName	Optional	Text	The SASFormatName. Business Rules: The SASFormatName must be a legal SAS format. The SASFormatName needs to start with a "$" character in case the CodeList DataType is "text".

5.3.12.1. EnumeratedItem Element

The EnumeratedItem element defines a CodedValue in a Controlled Terminology. Lists the CodedValues for all items in the controlled terminology.

Element Name:	EnumeratedItem
Element XPath:	/ODM/Study/MetaDataVersion/CodeList/EnumeratedItem
Element Textual Value:	*None*
Usage	Requirement: Conditional Cardinality: • Each CodeList element must contain either one or more EnumeratedItem elements, one or more CodeListItem elements or one ExternalCodelist element. Business Rules: • For Controlled Terminologies, where there is just a list of allowed values, an EnumeratedItem must be provided for each Item included in the Terminology. • The complete set of values relevant to the study must be provided whether or not they are referenced within the study data.
Attributes:	CodedValue, Rank, OrderNumber, def:ExtendedValue
Child Elements :	*Alias*

Attribute	Usage	Allowable Values	Description
CodedValue	Required	Text	The coded value. Business Rule: For NCI/CDISC Controlled Terminology, this must exactly match the CodedValue from the published Controlled Terminology ODM.
Rank	Optional	Integer	Numeric significance of the EnumeratedItem relative to others in the CodeList. Business Rule: If this value is provided for any EnumeratedItem, it must be provided for all. Note that the Rank attribute does not imply a display order.

OrderNumber	Optional	Integer	Display order of the item within the CodeList. Business Rule: If this value is provided for any *EnumeratedItem*, it must be provided for all.
def:ExtendedValue	Conditional Required when the CodedValue is an extended value.	Allowable Value: Yes	Indicates a coded value that has been used by the sponsor to extend external controlled terminology Note that Controlled Terminologies should only be extended by the sponsor in case the Controlled Terminology allows extension, and only in the case where there is no equivalent value or synonym already in the CodeList. Note that the attribute should be omitted when the CodedValue is not an extended value since the only allowable value is "Yes".

5.3.12.2. CodeListItem Element

The CodeListItem element defines a CodedValue in a Controlled Terminology when a Decode value or Preferred Term is provided for each code. It lists the Coded Values and Decodes for all items in the controlled terminology.

Element Name:	CodeListItem
Element Xpath:	/ODM/Study/MetaDataVersion/CodeList/CodeListItem
Element Textual Value:	*None*
Usage	Requirement: Conditional Cardinality: • Each CodeList element must contain either one or more EnumeratedItem elements, one or more CodeListItem elements or one ExternalCodelist element. Business Rules: • For Controlled Terminologies where there are Coded and Decoded values, a CodeListItem must be provided for each Item included in the Terminology. • The complete set of values relevant to the study must be provided whether or not they are referenced within the study data.
Attributes:	CodedValue, OrderNumber, Rank, def:ExtendedValue
Child Elements :	*Decode, Alias*

Attribute	Usage	Allowable Values	Description
CodedValue	Required	Text	The coded value. Business Rule: For NCI/CDISC Controlled Terminology, this must exactly match the CodedValue from the published Controlled Terminology ODM.
Rank	Optional	Integer	Numeric significance of the EnumeratedItem relative to others in the CodeList. Business Rule: If this value is provided, it must be present for all CodeListItems with the same parent CodeList element. Note that the Rank attribute does not imply a display order.

OrderNumber	Optional	Integer	Display order of the item within the CodeList. Business Rule: If this value is provided for any *CodeListItem*, it must be provided for all.
def:ExtendedValue	Conditional Required when the CodedValue is an extended value.	Allowable Value: Yes	Indicates a coded value that has been used by the sponsor to extend external controlled terminology Note that Controlled Terminologies should only be extended by the sponsor in case the Controlled Terminology allows extension, and only in the case where there is no equivalent value or synonym already in the CodeList. Note that the attribute should be omitted when the CodedValue is not an extended value since the only allowable value is "Yes".

5.3.12.3. Decode Element

The Decode element defines a preferred term for a CodedValue CodedValue in a CodeListItem.

Element Name:	Decode
Element Xpath:	/ODM/Study/MetaDataVersion/CodeList/CodeListItem/Decode
Element Textual Value:	*None*
Usage	Requirement: Required Cardinality: One Other Information: this element is the Container for Decode value, which is provided in the child element TranslatedText.
Attributes:	*None*
Child Elements :	*TranslatedText*

5.3.12.4. ExternalCodeList Element

Identifies the source of a 3rd party controlled terminology.

Element Name:	ExternalCodeList
Element Xpath:	/ODM/Study/MetaDataVersion/CodeList/ExternalCodeList
Element Textual Value:	*None*
Usage	Requirement: Conditional • For Controlled Terminologies provided by 3rd parties, an ExternalCodeList element must be provided to identify the Name and Version of the terminology. Cardinality: One • Each CodeList element must contain either one or more EnumeratedItem elements, one or more CodeListItem elements or one ExternalCodelist element. Business Rule: • Required for regulatory submissions to the FDA to provide the reference to the medical dictionaries used.
Attributes:	Dictionary, Version, ref, href
Child Elements :	*None*

Attribute	Usage	Allowable Values	Description
Dictionary	Required	Text	The name of the external codelist.
Version	Required	Text	The version designator of the external codelist.
Ref	Optional	Text	Reference to a local instance of the dictionary.
Href	Optional	Text	URL of an external instance of the dictionary.

5.3.13. MethodDef Element

A MethodDef element must be provided if any variables or values are defined as Derived.
One for each unique MethodRef@OID referenced.

Element Name:	MethodDef
Element Xpath:	/ODM/Study/MetaDataVersion/MethodDef
Element Textual Value:	*None*
Usage	Requirement: Conditional • Cardinality: Required for each unique value of the MethodOID attribute within the MetaDataVersion Business Rule: • Must contain the child Description element or the child def:DocumentRef element Other Information: • When the algorithm is provided in an External file the def:leafID attribute of the MethodDef element must be included and the Description element can include a short descriptive reference to the External file. • Note that each distinct method is expected to have a unique MethodOID and can be referenced from different variables.
Attributes:	OID, Name, Type
Child Elements :	Description, def:DocumentRef, FormalExpression

Attribute	Usage	Allowable Values	Attribute
OID	Required	Text	Unique ID for the MethodDef. See the ODM specification section 2.11 for OID considerations.
Name	Required	Text	The Method name.
Type	Required	Allowable Values: Computation Imputation	The Method type. A Computation uses an algorithm to derive a value. An Imputation is the process of replacing missing data with substitute values.

5.3.13.1. FormalExpression

Allows to provide machine readable code to compute or impute the value of an ItemDef.

Element Name:	FormalExpression
Element XPath:	/ODM/Study/MetaDataVersion/MethodDef/FormalExpression
Element Textual Value:	*text string*
Usage	Requirement: Optional Cardinality: Zero or More Business Rule: • The FormalExpression must evaluate to the correct DataType of the ItemDef that is to be imputed or computed using the Method in the computer language specified under Context. Other Information: • The way that the FormalExpression is to be combined with the rest of the code in the particular programming language specified in the Context attribute is outside of the scope of the Define-XML 2.0 specification.
Attributes:	Context
Child Elements :	*None*

Attribute	Usage	Allowable Values	Description
Context	Required	Text	A free-form qualifier to suggest an appropriate computer language to be used when evaluating the FormalExpression content.

5.3.14. def:CommentDef Element

The def:CommentDef element is new in Define-XML 2.0.0. It is intended to be used in place of the ODM Comment attribute. The use of the ODM Comment attribute within Define-XML files has been deprecated. The def:CommentOID attribute in ItemDef must reference a valid def:CommentDef. See section 5.3.11.

Element Name:	def:CommentDef
Element Xpath:	/ODM/Study/MetaDataVersion/def:CommentDef
Element Textual Value:	*None*
Usage	Requirement: Conditional Cardinality: Required for each unique value of the def:CommentOID attribute within the MetaDataVersion. Business Rule: • Must contain the child Description element or the child def:DocumentRef element Other Information: • When the comment is provided in an External file the def:leafID attribute of the def:CommentDef element must be included and the Description element can include a short descriptive reference to the External file. • Note that each distinct comment is expected to have a unique def:CommentOID and can be referenced from different variables.
Attributes:	OID
Child Elements :	Description, def:DocumentRef

Attribute	Usage	Allowable Values	Attribute
Attributes			
OID	Required	Text	Unique ID for the CommentDef. See the ODM specification section 2.11 for OID considerations.

5.3.15. def:leaf Element

Contains the Xlink information referenced by def:DocumentRef or def:ArchiveLocationID. Present if either the def:AnnotatedCRFID and/or the def:SupplementalDocID are provided.

Element Name:	def:leaf
Element Xpath:	/ODM/Study/MetaDataVersion/def:leaf /ODM/Study/MetaDataVersion/ItemGroupDef/def:leaf
Element Textual Value:	*None*
Usage	Requirement: Required Cardinality: One for each def:DocumentRef or ItemGroupDef included in the *define.xml* document
Attributes:	ID, xlink:href
Child Elements :	def:title

Attribute	Usage	Allowable Values	Description
ID	Required	Since the leaf ID is based on the XML xs:ID datatype, the allowed characters for the ID attribute are letters, digits, period, colons and hyphens.	Unique ID for the def:leaf. See the ODM specification section 2.11 for OID considerations. Business Rules: The def:leaf ID attributes must be unique within the *define.xml* document, i.e. there can be no 2 def:leaf elements with the same ID attribute.

Xlink:href	Required	Text	URL that can be used to identify the location of a document or dataset file relative to the folder containing the Define-XML file. If the file is not located in the Define-XML folder, a relative file path should be included. Business Rules: Currently, when referenced by def:ArchiveLocationID in the context of a regulatory submission not in ODM XML, it should be the pathname and filename of the SAS Transport file, including the ".xpt" extension, relative to the *define.xml* file. The value is used by the standard XSL stylesheet to provide a link to the corresponding SAS Transport file. For regulatory submissions to the FDA, the locations specified have to conform to locations allowed in the eCTD and the Study Data Specifications. See section 2.2 References.

5.3.15.1. Extended Child element def:title element

Element Name:	def:title
Element Xpath:	/ODM/Study/MetaDataVersion/def:leaf/def:title
Element Textual Value:	Text with the label for the document or dataset. It provides for a document title that differs from the file name, as may be found with annotated CRFs or supplemental documents.
Usage	Requirement: Required Cardinality: One Other Information: Note that the def:title element has no attributes or child elements.
Attributes:	*None*
Child Elements :	*None*

6. Global Element Ordering

Following the order of elements is a part of conformance to the Define-XML, therefore, all of the elements are listed below in their correct order for your reference.

```
<ODM>
  <Study>
  <GlobalVariables>
    <StudyName>
    <StudyDescription>
    <ProtocolName>
  <MetaDataVersion>
    <def:AnnotatedCRF>
      <def:DocumentRef>
    <def:SupplementalDoc>
      <def:DocumentRef>
    <def:ValueListdef>
      <ItemRef>
        <def:WhereClauseRef>
    <def:WhereClauseDef>
      <RangeCheck>
        <CheckValue>
    <ItemGroupDef>
      <Description>
        <TranslatedText>
      <ItemRef>
      <def:leaf>
        <def:title>
    <ItemDef>
      <Description>
        <TranslatedText>
      <CodeListRef>
      <def:Origin>
        <def:DocumentRef>
          <def:PDFPageRef>
      <def:ValueListRef>
    <CodeList>
      <EnumeratedItem>
        <Alias>
      <CodeListItem>
        <Decode>
          <TranslatedText>
        <Alias>
      <ExternalCodeListItem>
      <Alias>
    <MethodDef>
      <Description>
        <TranslatedText>
      <def:DocumentRef>
        <def:PDFPageRef>
    <def:CommentDef>
      <Description>
        <TranslatedText>
      <def:DocumentRef>
        <def:PDFPageRef>
    <def:leaf>
      <def:title>
```

7. Acknowledgments

This specification was developed by the CDISC Define-XML Development Team:

- Sam Hume, AstraZeneca
- Sally Cassells, Next Step Clinical Systems LLC
- Kevin Burges, Formedix
- Lex Jansen, SAS Institute
- Marcelina Hungria / DIcore Group, LLC
- Mike Molter, D-Wise
- Jozef Aerts, XML4Pharma
- Jan Kratky, SAS Institute

8. Changes from Previous Version

The major changes between this version and the published V1.0.0 version are described in an accompanying PDF file named 'Define2-NewFeatures.PDF'.

9. Deprecated Components

Some components of define-XML v1 have been deprecated. A list of the deprecated elements and attributes is provided in table below.

Element(s)	Component	Comment
MetaDataVersion	def:ComputationalMethod	Replaced by ODM 1.3 MetaDataVersion/MethodDef element.
ItemGroupDef	def:Label	Replaced by ODM 1.3 ItemGroupDef/Description element.
ItemGroupDef	Def:DomainKeys	Replaced by ODM 1.3 ItemGroupDef/ItemRef/@KeySequence attribute.
ItemDef	def:Label	Replaced by ODM 1.3 ItemDef/Description element.
ItemDef	def:ComputationMethodOID	Replaced by ODM 1.3 ItemRef/@MethodOID attribute
ItemDef	Origin	Replaced by new ItemDef/def:Origin element.
ItemDef	Comment	Replaced by ItemDef/@def:CommentOID attribute and MetaDataVersion/def:CommentDef element
CodeListItem	def:Rank	Replaced by ODM 1.3 attribute Rank.

Appendix 1: XML Schema

The examples in this document are included in an XML file as part of the DefineXML 2.0 publication. This XML file references (directly or indirectly) the following schema files:

DefineXML schema	schema/cdisc-define-2.0 (top level folder)
	schema/cdisc-define-2.0/define2-0-0.xsd
	schema/cdisc-define-2.0/define-extension.xsd
	schema/cdisc-define-2.0/define-ns.xsd
ODM 1-3-2 Schema	schema/cdisc-odm-1.3.2 (top level folder)
	schema/cdisc-odm-1.3.2/ODM1-3-2.xsd
	schema/cdisc-odm-1.3.2/ODM1-3-2-foundation.xsd
	xlink.xsd
	xml.xsd
	xmldsig-core-schema.xsd

Appendix 2: Visualizing Value Level Metadata

Viewing Value Level Metadata as Value Lists

A Value List describes all of the possible definitions of the contents of a variable when different conditions hold true, for example different definitions of VSORRES when VSTESTCD = 'TEMP' or when VSTESTCD = 'HR'.

As an example, in the Vital Signs domain the variables VSORRES, VSORRESU are defined as shown below.

Name	Label	Type	Controlled Terms	Role
VSORRES	Result or Finding in Original Units	text		Result Qualifier
VSORRESU	Original Units	text	Units for Vital Signs Results (C66770)	Variable Qualifier

The Type and Controlled Terms for VSORRES and VSORRESU can differ for each test, so Value Level Metadata can be used to indicate the definition of the Variable for each different VSTESTCD Value. A Value List should be attached to each variable whose definition changes depending on the values of another dataset variable.

Source Variable	Condition	Label	Type	Controlled Terms
VSORRES	VSTESTCD EQ 'TEMP'	Temperature	float	
VSORRES	VSTESTCD EQ 'HR'	Heart Rate	integer	

Source Variable	Condition	Label	Type	Controlled Terms
VSORRESU	VSTESTCD EQ 'TEMP'	Temperature	text	['C', 'F']
VSORRESU	VSTESTCD EQ 'HR'	Heart Rate	text	['BPM']

This shows how VSORRES and VSORRESU are defined for each different test.

The above example shows a simple Value List where conditions that define the different Values are simply "Where VSTESTCD EQ '<value>'". The conditions can be more complicated, e.g. "Where VSTESTCD EQ '<value>' and VSCAT EQ '<value>'".

Note: The tables in this section are just illustrative. They do not define how Value Lists should be displayed.

Viewing Value Level Metadata as Slices

Slices are also defined by Value Lists but they are displayed in a way that illustrates the domain dataset point ofo view. Instead of showing all the possible Values of a Variable under different conditions as a Value List does, a Slice shows what a dataset looks like under a specific condition. Using the example above, the VS domain could be shown as two Slices, one where VSTESTCD = 'TEMP' (or a "TEMP slice") and one where VSTESTCD = 'HR' (or a "HR slice"). This can be shown as below.

TEMP Slice: VS Domain Where VSTESTCD EQ 'TEMP'

Name	Label	Type	Controlled Terms
STUDYID	Study Identifier	text	
DOMAIN	Domain Abbreviation	text	['VS']
USUBJID	Unique Subject Identifier	text	
VSSEQ	Sequence Number	float	
VSTESTCD	Vital Signs Test Short Name	text	Vital Signs Test Code (C66741)
VSTEST	Vital Signs Test Name	text	Vital Signs Test Name (C67153)
VSORRES	Temperature	float	
VSORRESU	Temperature	text	['C']
VSSTRESC	Character Result/Finding in Std Format	text	
VSSTRESN	Numeric Result/Finding in Standard Units	float	
VSSTRESU	Standard Units	text	Units for Vital Signs Results (C66770)
VSBLFL	Baseline Flag	text	No Yes Response (C66742)
VISITNUM	Visit Number	float	
VSDTC	Date/Time of Measurements	datetime	ISO 8601 (Dates/Times)

HR Slice: VS Domain Where VSTESTCD EQ 'HR'

Name	Label	Type	Controlled Terms
STUDYID	Study Identifier	text	
DOMAIN	Domain Abbreviation	text	['VS']
USUBJID	Unique Subject Identifier	text	
VSSEQ	Sequence Number	float	
VSTESTCD	Vital Signs Test Short Name	text	Vital Signs Test Code (C66741)
VSTEST	Vital Signs Test Name	text	Vital Signs Test Name (C67153)
VSORRES	Heart Rate	integer	
VSORRESU	Heart Rate	text	['BPM']
VSSTRESC	Character Result/Finding in Std Format	text	
VSSTRESN	Numeric Result/Finding in Standard Units	float	
VSSTRESU	Standard Units	text	Units for Vital Signs Results (C66770)
VSBLFL	Baseline Flag	text	No Yes Response (C66742)
VISITNUM	Visit Number	float	
VSDTC	Date/Time of Measurements	datetime	ISO 8601 (Dates/Times)

Most Variables will share a common definition across all Slices, e.g. –DOMAIN, –SEQ. Only the Variables which have a separate definition for each Slice need be displayed separately for each Slice. If a Variable does not have a definition specific to a given Slice (i.e., does not have a Value definition for that Slice), the definition of the parent Variable is used. This minimizes the amount of metadata required to define Slices.

Note: The tables in this section are just illustrative. They do not define how Slices should be shown.