PhUSE 2011

Paper CD06

Using the SAS® Clinical Standards Toolkit to work with the CDISC ODM model

Lex Jansen, SAS Institute Inc., Cary, North Carolina, USA

ABSTRACT

The CDISC Operational Data Model (ODM) is a vendor neutral, platform independent format for the interchange and archival of clinical study data. The model represents study metadata, administrative metadata, reference data and subject data associated with a clinical trial. The ODM format is defined by an XML schema and a specification.

The SAS® Clinical Standards Toolkit (CST) is a framework of SAS macros, metadata and configuration files to support clinical research activities. The CDISC ODM standard is one of the standards that are supported by CST. To enable ODM support, SAS has defined a relational data model that represents the ODM model as SAS data sets. This paper shows how the SAS Clinical Standards Toolkit supports:

- Reading and writing of ODM files.
- Schema level validation of an ODM file
- Validating structure and content of the SAS representation of ODM

The paper also looks at the inner workings and underlying technologies of the SAS Clinical Standards Toolkit with respect to XML based standards.

INTRODUCTION

This paper discusses the upcoming release 1.4 of the SAS Clinical Standards Toolkit [1]. The SAS Clinical Standards Toolkit 1.4 is supported with SAS 9.3. This release of the Toolkit supports reading and writing of files that conform to the Operational Data Model standard version 1.3.0.

SAS® CLINICAL STANDARDS TOOLKIT

The SAS Clinical Standards Toolkit (CST) focuses primarily on supporting clinical research activities. These activities involve the discovery and development of new pharmaceutical and biotechnology products and medical devices. These activities occur from project initiation through product submission and throughout the full product lifecycle.

The SAS Clinical Standards Toolkit initially focuses on standards defined by the Clinical Data Interchange Standards Consortium (CDISC). CDISC is a global, open, multidisciplinary, nonprofit organization that has established standards to support the acquisition, exchange, submission, and archival of clinical research data and metadata. The CDISC mission is to develop and support global, platform-independent data standards that enable information-system interoperability, which, in turn, improves medical research and related areas of health care. The SAS Clinical Standards Toolkit is not limited to supporting CDISC standards. In time, the SAS Clinical Standards Toolkit will support other evolving industry-standard data models. The SAS Clinical Standards Toolkit framework is designed to support the specification and use of any user-defined standard.

The term "toolkit" connotes a collection of tools, products, and solutions. The SAS Clinical Standards Toolkit provides a set of standards and functionality that will evolve and grow with future product updates and releases. Customer requirements and expectations of the SAS Clinical Standards Toolkit will play a key role in the deciding what functionality to provide in future releases.

The SAS Clinical Standards Toolkit 1.4 provides support for the following CDISC standards:

- SDTM 1.3.1 and 3.1.2
- ADaM 2.1 (ADSL, Basic Data Structure and Analysis Results Metadata templates, as well as version 1.1 of the ADaM validation checks)
- CRT-DDS 1.0 (define.xml)
- ODM 1.3.0

- CDISC Controlled Terminology package that includes the cumulative set of terminology as posted to the NCI FTP site as of April 2011

Each SAS Clinical Standards Toolkit standard provides a SAS representation of the published source guidelines or source specification. The SAS representation is designed to serve as a model or template of the source specification. Two key design requirements shaped the implementation of the SAS Clinical Standards Toolkit standards.

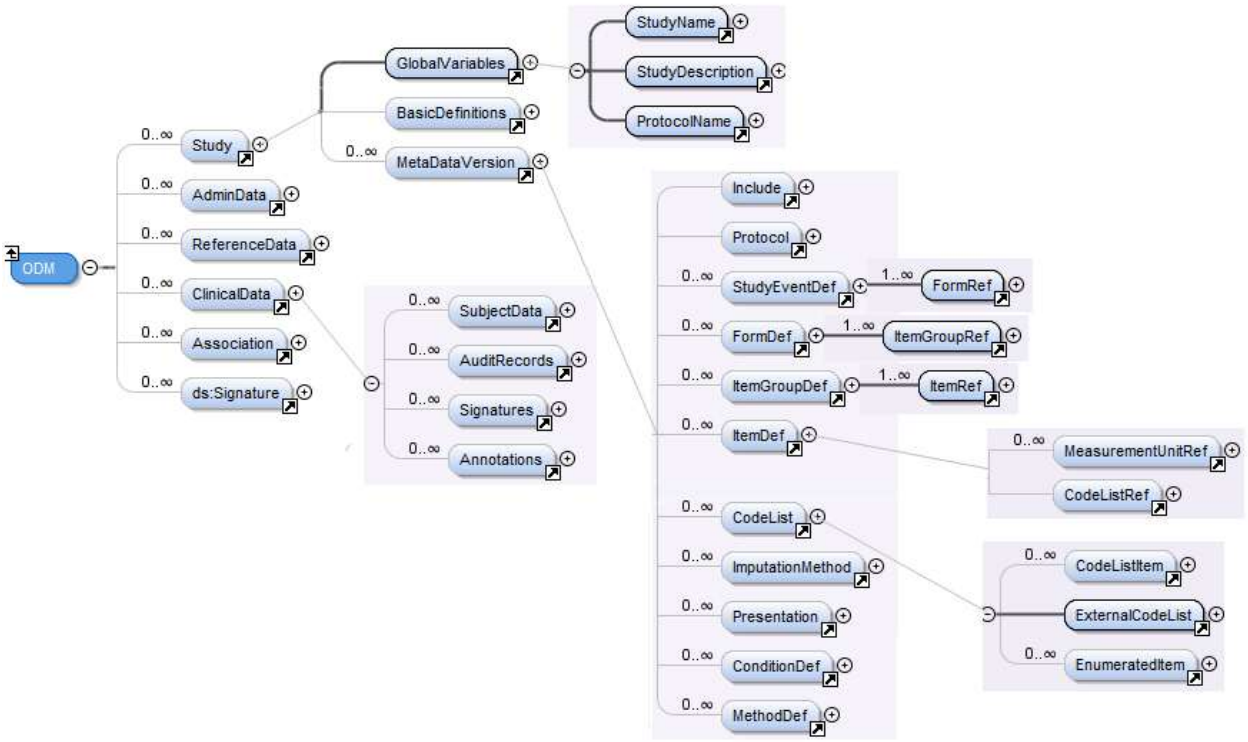
1) Each supported standard is represented in one or more SAS files. This facilitates these points:
   - It provides SAS users with an implementation of data models and standards that are based on SAS.
   - It enables you to use SAS routines to assess how well any user-defined set of data and metadata conforms to the standard.
   - It enables you to use SAS code to read and derive files in other formats (for example, XML).

   Each SAS Clinical Standards Toolkit standard is an optimized reference standard from a SAS perspective.
2) You are able to define your own customized standards, or you are able to modify existing SAS standards.

OPERATIONAL DATA MODEL (ODM)

The CDISC Operational Data Model (ODM) [2] is a vendor neutral, platform independent XML based format for the interchange and archival of clinical study data. The model represents study metadata, administrative metadata, reference data and subject data associated with a clinical trial. The ODM format is defined by an XML schema and a specification. Figure 1 partially displays the ODM 1.3.0 element structure. For a full understanding of the ODM structure one has to study the ODM specification and the XML schema that defines ODM 1.3.0.

Figure 1 ODM: partial structure



SAS® CLINICAL STANDARDS TOOLKIT AND ODM

SAS® CLINICAL STANDARDS TOOLKIT AND ODM

The SAS Clinical Standards Toolkit 1.4 provides the following support of the CDISC ODM 1.3.0 standard:

- reading and representing in SAS a complete ODM XML file (specific limitations are noted below)
- building an odm.xml file from a SAS representation of the ODM standard
- schema-level validation of an odm.xml file

- validating the structure and content of the SAS representation of an odm.xml file
- A sample tool to identify unsupported ODM elements and attributes.

The SAS Clinical Standards Toolkit 1.4 will NOT support the following CDISC ODM 1.3.0 functionality:

- Reading or writing the DigitalSignatures section of the ODM
- Vendor or customer extensions of the ODM
- Any functionality added with ODM 1.3.1 (will be added in the future)
- Processing is limited to a single ODM file (i.e., use of PriorFileOID to reference another file is ignored).
- Full file metadata is expected in each file.
- Effective support only for ODM FileType="Snapshot". The SAS Clinical Standards Toolkit 1.4 makes no attempt to process multiple transactions per data point; multiple transactions are saved in the SAS ODM representation for subsequent processing

The SAS PROC CDISC procedure supports bidirectional conversion of data content contained in a CDISC ODM XML document to and from data sources that are accessible through SAS. However, PROC CDISC supports CDISC ODM version 1.2 only. PROC CDISC is maintenance only – no new features will be added.
The SAS Clinical Standards Toolkit is in active development.

With the production releases of the SAS Clinical Standards Toolkit 1.4, the CDISC ODM reference standard will support reading and representing in SAS a complete odm.xml file, building an odm.xml file, and validating the structure and content of the SAS representation of an odm.xml file. In addition, it will validate the structural integrity of the ODM XML file. To support this functionality, supplemental files include the following global standards library files:

- A SAS format catalog (odmfmtcat*.sas7bcat) in the formats folder provides valid values for selected columns in the 66 tables of the SAS representation.
- The Messages data set in the messages folder provides error messaging for all Validation Master checks.
- The Validation Master data set in the validation/control folder contains the super-set of checks validating the structure and content of the 66 tables
- SAS code in the macros folder provides CDISC ODM-specific code that augments code that is provided in the primary SAS Clinical Standards Toolkit autocall library (!sasroot/cstframework/sasmacro).

It is this set of files, in whole or in part, that defines the CDISC ODM reference standard in SAS.

SAS® CLINICAL STANDARDS TOOLKIT AND ODM – THE SAS DATA MODEL FOR ODM

The highly-structured nature of ODM data requires that any mapping to a relational format includes a large number of data sets, with enforcement of foreign-key relationships to help preserve the intended non-relational object structure. In the SAS Clinical Standards Toolkit, key relationships will be enforced when validating the ODM datasets.

Field lengths in the CDISC ODM data sets are consistent by core data type. CDISC has not specified any limit to the length of most character fields. Arbitrary lengths have been chosen by data type. These lengths are listed in the following table. In the table, standard data types are distilled into core data types. To be safe, larger lengths have been chosen to ensure that no data loss occurs in the SAS Clinical Standards Toolkit pre-installed data sets. Production tables might be compressed using SAS mechanisms to preserve disk space.

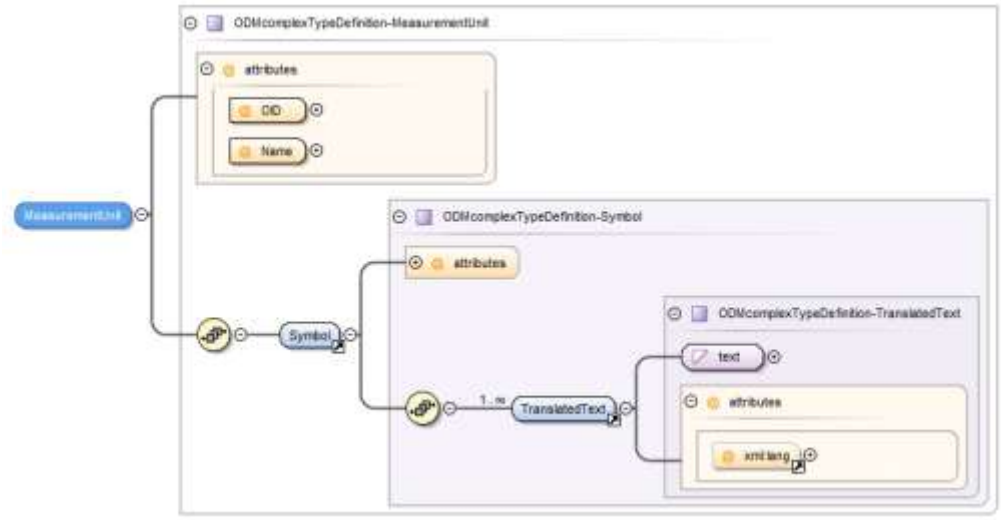
Table 1 CDISC ODM Default Lengths by Data Type

Type Name	Length	Description
oid	64	A unique object identifier or a reference
text	2000	A character field that can accommodate a large number of characters
name	128	A descriptive identifier
value	512	An item of collected or reference data
path	512	An absolute or relative file system path or URL

When mapping the ODM XML schema to SAS data sets, in general, elements become SAS data sets and attributes become columns in the data sets. When there is a 1-n relation between an XML element and its sub-element, the element and its sub-element become different SAS data sets.
The example in Figure 2 illustrates this concept. The physical unit of measure for a data item or value has a unique identifier (OID) and a name (Name) and a symbol providing a human-readable name for the measurement unit that is appropriate for a particular language. TranslatedText elements typically occur in a series, presenting a set of alternative textual renditions for different languages. To avoid ambiguity, a particular language tag must not occur more than once in a series of TranslatedText elements. Also, it is not permitted to have more than one

TranslatedText element without an xml:lang attribute within the same parent.
The MeasurementUnit element and its attributes and TranslatedText sub-element are mapped to 2 SAS data sets: MeasurementUnits and MuTranslatedText.

Figure 2 ODM to SAS data set mapping




```
<BasicDefinitions>
  <MeasurementUnit OID="MeasurementUnits.OID.BEAT_MIN" Name="BEAT_MIN">
    <Symbol>
      <TranslatedText xml:lang="en">beats/minute</TranslatedText>
      <TranslatedText xml:lang="fr-CA">bat/minute</TranslatedText>
    </Symbol>
  </MeasurementUnit>
  <MeasurementUnit OID="MeasurementUnits.OID.CM" Name="CM">
    <Symbol>
      <TranslatedText xml:lang="en">CM</TranslatedText>
      <TranslatedText xml:lang="fr-CA">CM</TranslatedText>
    </Symbol>
  </MeasurementUnit>
  <MeasurementUnit OID="MeasurementUnits.OID.MMHG" Name="MMHG">
    <Symbol>
      <TranslatedText xml:lang="en">mmHG</TranslatedText>
      <TranslatedText xml:lang="fr-CA">mmHG</TranslatedText>
    </Symbol>
  </MeasurementUnit>
  <MeasurementUnit OID="MeasurementUnits.OID.YRS" Name="YEARS">
    <Symbol>
      <TranslatedText xml:lang="de">Jahren</TranslatedText>
      <TranslatedText xml:lang="en">Years of age</TranslatedText>
      <TranslatedText xml:lang="fr-CA">Ans</TranslatedText>
    </Symbol>
  </MeasurementUnit>
</BasicDefinitions>
```

VIEWTABLE: Srcdata.Measurementunits

	OID	Name	FK_Study
1	MeasurementUnits.OID.BEAT_MIN	BEAT_MIN	Study.OID
2	MeasurementUnits.OID.CM	CM	Study.OID
3	MeasurementUnits.OID.MMHG	MMHG	Study.OID
4	MeasurementUnits.OID.YRS	YEARS	Study.OID

VIEWTABLE: Srcdata.Mutranslatedtext

	TranslatedText	lang	FK_MeasurementUnits
1	beats/minute	en	MeasurementUnits.OID.BEAT_MIN
2	bat/minute	fr-CA	MeasurementUnits.OID.BEAT_MIN
3	CM	en	MeasurementUnits.OID.CM
4	CM	fr-CA	MeasurementUnits.OID.CM
5	mmHG	en	MeasurementUnits.OID.MMHG
6	mmHG	fr-CA	MeasurementUnits.OID.MMHG
7	Jahren	de	MeasurementUnits.OID.YRS
8	Years of age	en	MeasurementUnits.OID.YRS
9	Ans	fr-CA	MeasurementUnits.OID.YRS

The complete listing of the SAS data sets that comprise the ODM 1.3.0 standard can be seen in Appendix 1.

SAS® CLINICAL STANDARDS TOOLKIT AND ODM – PROCESSING XML

The SAS Clinical Standards Toolkit uses an intermediate XML format when transforming an ODM XML file into SAS data sets or transforming SAS data sets into an ODM XML file. This intermediate format is a very 'flat' representation of the table data in XML and will consist of extremely simple markup describing the tables and column data.

This very simple XML 'cube' format captures all tabular data and is sufficient as input to the XML transformation step, which will consume the 'cube' XML file and produce a standards-compliant ODM XML file. Figure 3 shows the overall structure of the XML 'cube' document.

Figure 3 XML 'cube'

```
<LIBRARY>
    <TableName0>
        <Column0>some data</Column0>
        <Column1>some data</Column1>
        <Column2>some data</Column2>
    </TableName0>
    <TableName1>
        <AColumn0>some data</AColumn0>
        <AColumn1>some data</AColumn1>
    </TableName1>
    <TableName1>
        <AColumn0>some data</AColumn0>
        <AColumn1/>
    </TableName1>
</LIBRARY>
```

We refer to this sort of XML representation of relational table data as a 'cube' XML format, so named because the XML is three levels (or 'dimensions') deep:

- All XML documents must have exactly one root element. In our format, the root element will always be named 'LIBRARY', and will have no additional attributes.
- Each child element of the root will have as its name the name of a dataset. Each occurrence of such an element represents a single row/observation in that dataset. Therefore, if a dataset contains more than one row of data, multiple elements having the same name will exist at this level. Although the order in which distinct dataset names appear in this file is unimportant, all elements representing rows in the same dataset MUST appear consecutively. Dataset name case must be preserved (i.e. dataset names cannot be produced in all caps).
- The third, and deepest, level of elements in the XML document represents the rows within the dataset represented by the parent of each. The name of each element at this level is the same as the name (case-sensitive) of the SAS dataset column. For each observation, exactly one element must be produced for each row in the dataset, regardless of whether there is data in that column for the row currently being represented. A cell that contains no data will be represented as an empty XML element, as shown above with the <AColumn1/> element. Otherwise, the element will contain the character data from the corresponding dataset cell, exactly as it had been represented in the staging datasets.

Figure 4 shows an example of this XML 'cube' representing the MeasurementUnits example as we already saw in Figure 2.

Figure 4 XML 'cube' example

```
    <MeasurementUnits>
        <OID>MeasurementUnits.OID.BEAT_MIN</OID>
        <Name>BEAT_MIN</Name>
        <FK_Study>Study.OID</FK_Study>
    </MeasurementUnits>
    <MeasurementUnits>
        <OID>MeasurementUnits.OID.CM</OID>
        <Name>CM</Name>
        <FK_Study>Study.OID</FK_Study>
    </MeasurementUnits>
    <MeasurementUnits>
        <OID>MeasurementUnits.OID.MMHG</OID>
        <Name>MMHG</Name>
        <FK_Study>Study.OID</FK_Study>
    </MeasurementUnits>
    <MeasurementUnits>
        <OID>MeasurementUnits.OID.YRS</OID>
```

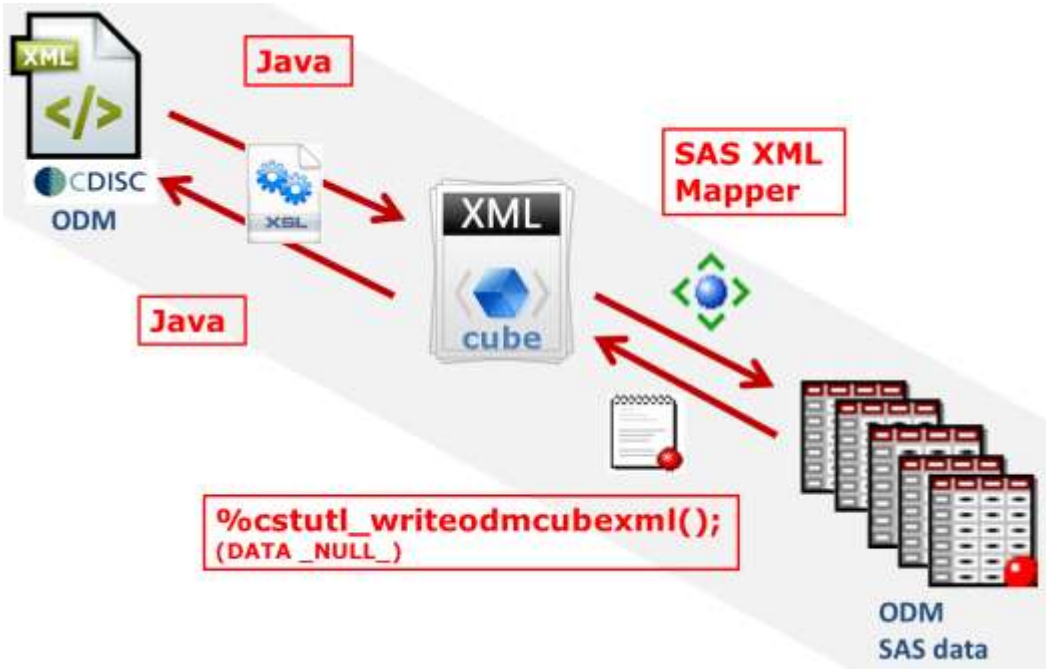
```
    <Name>YEARS</Name>
    <FK_Study>Study.OID</FK_Study>
</MeasurementUnits>
<MUTranslatedText>
    <TranslatedText>beats/minute</TranslatedText>
    <lang>en</lang>
    <FK_MeasurementUnits>MeasurementUnits.OID.BEAT_MIN</FK_MeasurementUnits>
</MUTranslatedText>
<MUTranslatedText>
    <TranslatedText>bat/minute</TranslatedText>
    <lang>fr-CA</lang>
    <FK_MeasurementUnits>MeasurementUnits.OID.BEAT_MIN</FK_MeasurementUnits>
</MUTranslatedText>
```

A Java process is invoked (through the SAS data step JavaObj component) that leverages XSL technology to transform between the 'cube' XML and the ODM XML file. The SAS Clinical Standards Toolkit uses the Xerces Java parser to validate the ODM XML file and to obtain informational messages about the validation result.

The SAS macro **cstutl_writeodmcubexml** transforms the SAS data sets that represent an ODM XML file to the 'cube' XML file by using a simple DATA _NULL_ step. By using a DATA _NULL_ step instead of the SAS XML engine we do not have to make any data type inferences. This DATA _NULL_ step is driven by the tables and columns metadata that define the ODM standard in the SAS Clinical Standards Toolkit.

The SAS data sets that represent the ODM XML file can be created from the 'cube' XML file by utilizing SAS XML Mapper technology. The XML Map used by the SAS XML Mapper is completely determined by the metadata that describes the SAS representation of the ODM XML standard, i.e. the *reference_tables* and *reference_columns* SAS data sets . Figure 5 gives an overview of the various transformations and the technologies used.

Figure 5 Transformations between ODM XML and SAS



SAS® CLINICAL STANDARDS TOOLKIT – THE GENERIC PROCESS

The SAS Clinical Standards Toolkit Global Library contains the SAS metadata definition of all supported standards. This metadata definition, defined primarily as the *reference_tables* and *reference_columns* data sets, can be used as defined or may be modified or used as a template to build your own SAS representation of the SAS-supplied CDISC standards or your own customized standard. Sample driver modules are provided with each standard that is supported by the SAS Clinical Standards Toolkit. These driver modules can be copied and modified to reflect your own data and metadata sources and the target location for any SAS process output. These drivers all follow the same general process workflow:

- Set any process global macro variable values.
  Global macro variables are set by utilizing a so-called properties file that has name-value pairs.
- Define a root path for input and output files
- Create or reference a *SASReferences* data set that defines all input/output files
- Call the **cstutil_processsetup** macro that confirms a valid *SASReferences* structure, allocates any SAS librefs and filerefs, set the process-specific macro autocall and format search paths
- Call the primary macro that performs the task of interest (such as validation or reading/writing an ODM XML file)
- Optionally perform any session cleanup

Perhaps the single most important file used by the SAS Clinical Standards Toolkit is the *SASReferences* data set. The *SASReferences* data set captures all the input and output file library and file references associated with any given process.
Figures 5a, 5b and 5c give examples of *SASReference* tables for various ODM 1.3.0 related processes.

Figure 5a Sample *SASReferences* SAS data set (ODM 1.3.0 Read ODM XML Process)

VIEWTABLE: Work.Sasreferences

standard	type	subtype	SASref	reftype	path	memname
CST-FRAMEWORK	messages		messages	libref		
CDISC-ODM	messages		odmmsg	libref		
CDISC-ODM	autocall		odmcode	fileref		
CDISC-ODM	sourcedata		srcdata	libref	!sasroot/../../SASClinicalStandardsToolkitODM130/1.4/sample/cdisc-odm-1.3.0/derived/data	
CDISC-ODM	sourcemetadata	table	srcmeta	libref	!sasroot/../../SASClinicalStandardsToolkitODM130/1.4/sample/cdisc-odm-1.3.0/derived/metadata	source_tables.sas7bdat
CDISC-ODM	sourcemetadata	column	srcmeta	libref	!sasroot/../../SASClinicalStandardsToolkitODM130/1.4/sample/cdisc-odm-1.3.0/derived/metadata	source_columns.sas7bdat
CDISC-ODM	targetdata		trgdata	libref	!sasroot/../../SASClinicalStandardsToolkitODM130/1.4/sample/cdisc-odm-1.3.0/derived/formats	
CDISC-ODM	referencemetadata	table	refmeta	libref		
CDISC-ODM	referencemetadata	column	refmeta	libref		
CDISC-ODM	results	results	results	libref	!sasroot/../../SASClinicalStandardsToolkitODM130/1.4/sample/cdisc-odm-1.3.0/results	read_results.sas7bdat
CDISC-ODM	externalxml	xml	odmxml	fileref	!sasroot/../../SASClinicalStandardsToolkitODM130/1.4/sample/cdisc-odm-1.3.0/sourcexml	odm_sample.xml
CDISC-ODM	referencexml	map	odmmap	fileref	!sasroot/../../SASClinicalStandardsToolkitODM130/1.4/sample/cdisc-odm-1.3.0/referencexml	odm.map
CDISC-ODM	properties	initialize	inprop	fileref		initialize.properties

Figure 5b Sample *SASReferences* SAS data set (ODM 1.3.0 Validation Process)

VIEWTABLE: Work.Sasreferences

standard	type	subtype	SASref	reftype	path	memname
CST-FRAMEWORK	messages		messages	libref		
CDISC-ODM	messages		odmmsg	libref		
CDISC-ODM	autocall		auto1	fileref		
CDISC-ODM	control	reference	cntl_s	libref	C:\Users\frjans\AppData\Local\Temp\SAS Temporary Files_TD6816_L72371_	sasreferences
CDISC-ODM	control	validation	cntl_v	libref	!sasroot/../../SASClinicalStandardsToolkitODM130/1.4/sample/cdisc-odm-1.3.0/control	validation_control
CDISC-ODM	fmtsearch		odmfmt	libref		
CDISC-ODM	referencecontrol	validation	refcntl	libref		
CDISC-ODM	referencemetadata	table		libref		
CDISC-ODM	referencemetadata	column		libref		
CDISC-ODM	sourcemetadata	table	srcmeta	libref	!sasroot/../../SASClinicalStandardsToolkitODM130/1.4/sample/cdisc-odm-1.3.0/metadata	source_tables
CDISC-ODM	sourcemetadata	column	srcmeta	libref	!sasroot/../../SASClinicalStandardsToolkitODM130/1.4/sample/cdisc-odm-1.3.0/metadata	source_columns
CDISC-ODM	sourcedata		srcdata	libref	!sasroot/../../SASClinicalStandardsToolkitODM130/1.4/sample/cdisc-odm-1.3.0/data	
CDISC-ODM	results	validationresults	results	libref	!sasroot/../../SASClinicalStandardsToolkitODM130/1.4/sample/cdisc-odm-1.3.0/results	validation_results
CDISC-ODM	results	validationmetrics	results	libref	!sasroot/../../SASClinicalStandardsToolkitODM130/1.4/sample/cdisc-odm-1.3.0/results	validation_metrics
CDISC-ODM	properties	initialize	inprop	fileref		initialize.properties
CDISC-ODM	properties	validation	valprop	fileref		validation.properties

Figure 5c Sample *SASReferences* SAS data set (ODM 1.3.0 Create ODM XML Process)

VIEWTABLE: Work.Sasreferences

standard	type	subtype	SASref	reftype	path	memname
CST-FRAMEWORK	messages		messages	libref		
CDISC-ODM	messages		odmmsg	libref		
CDISC-ODM	autocall		auto1	fileref		
CDISC-ODM	control	reference	control	libref	C:\Users\frjans\AppData\Local\Temp\SAS Temporary Files_TD6816_L72371_	sasreferences
CDISC-ODM	results	results	results	libref	!sasroot/../../SASClinicalStandardsToolkitODM130/1.4/sample/cdisc-odm-1.3.0/results	write_results.sas7bdat
CDISC-ODM	sourcedata		srcdata	libref	!sasroot/../../SASClinicalStandardsToolkitODM130/1.4/sample/cdisc-odm-1.3.0/data	
CDISC-ODM	externalxml	xml	extxml	fileref	!sasroot/../../SASClinicalStandardsToolkitODM130/1.4/sample/cdisc-odm-1.3.0/sourcexml	odm_sample_out.xml
CDISC-ODM	properties	initialize	inprop	fileref		initialize.properties

Some general observations about the sample *SASReferences* data set shown above:

- The type and subtype values are used by the SAS Clinical Standards Toolkit code to find the fileref or libref associated with a particular input or output file type.
- The SASref may be any user-defined value.
- An empty path tells the SAS Clinical Standards Toolkit to look for the default value from the standard-specific metadata in the Global Library (usually a Global Library location).

SAS® CLINICAL STANDARDS TOOLKIT AND ODM – READING ODM

The SAS code for reading an ODM XML file is given below in a shortened form:

```
%let _cstStandard=CDISC-ODM;
%let _cstStandardVersion=1.3.0;
%let _cstVersion=;

%cst_setStandardProperties(_cstStandard=CST-FRAMEWORK,_cstSubType=initialize);

%cst_createds(_cstStandard=CST-FRAMEWORK, _cstType=control,_cstSubType=reference,
_cstOutputDS=work.sasreferences);

proc sql;
  insert into work.sasreferences
  values("CST-FRAMEWORK"  "1.2"  "messages"  ""  "messages"  "libref"  ""  1  ""  "")
  values("& cstStandard"  "& cstStandardVersion"  "externalxml"  "xml"  "odmxml"  "fileref"
         "&studyRootPath/sourcexml"  1  "odm_sample.xml"  "")
…
  ;
quit;

%cstutil_processsetup();
%odm_xmlvalidate();
%odm_read();
```

Running this code will result in 66 SAS data sets which contain the SAS representation of the ODM XML file. The inputs and outputs of this macro are specified via a *SASReferences* file. For the complete code the reader is referred to the sample driver program in the sample study that ships with the SAS Clinical Standards Toolkit. The **odm_read** macro has several optional macro parameters that can be specified.
Some of these parameters:

_cstBuildSrcMetadata	Create the source metadata files (e.g. source_tables and source_columns) as a part of the read operation. Default=Y (yes)
_cstBuildFmtCat	Build format catalog(s), representing language-specific codelist TranslatedText, as a part of the read operation. Default=Y (yes)

SAS® CLINICAL STANDARDS TOOLKIT AND ODM – WRITING ODM

The SAS code for creating an ODM XML file is given below in a shortened form:

```
%let _cstStandard=CDISC-ODM;
%let _cstStandardVersion=1.3.0;
%let _cstVersion=;

%cst_setStandardProperties(_cstStandard=CST-FRAMEWORK,_cstSubType=initialize);

%cst_createds(_cstStandard=CST-FRAMEWORK, _cstType=control,_cstSubType=reference,
_cstOutputDS=work.sasreferences);

proc sql;
  insert into work.sasreferences
  values("CST-FRAMEWORK"  "1.2"  "messages"  ""  "messages"  "libref"  ""  1  ""  "")
  values("& cstStandard"  "& cstStandardVersion"  "externalxml"  "xml"  "odmxml"  "fileref"
         "&studyOutputPath/sourcexml"  1  "odm_sample_out.xml"  "")
…
  ;
quit;

%cstutil_processsetup();
```

```
%odm_write();
%odm_xmlvalidate();
```

Running this code will result in an ODM XML file created from the 66 SAS data sets which contain the SAS representation of the ODM XML file. The inputs and outputs of this macro are specified via a *SASReferences* file. For the complete code the reader is referred to the sample driver program in the sample study that ships with the SAS Clinical Standards Toolkit. The **odm_write** macro has several optional macro parameters that can be specified.

_cstCreateDisplayStyleSheet	Identifies whether the macro should create a stylesheet in the same directory as the output XML file. If this is set to 1, the macro will look in the supplied SASreferences file for a record with type/subtype of referencexml/stylesheet and will use that file. If set to 0, the macro will not create the XSL, even if one is specified in the SASReferences file. Default=0. A stylesheet is typically not used for an ODM XML file.
_cstOutputEncoding	The XML encoding to use for the ODM file that is created. Default=UTF-8.
_cstHeaderComment	A short comment will be added to top of the ODM XML file that is produced. If none is provided, a default will be used.
_cstResultsOverrideDS	The (libname.)member that refers to a results data set to be created. If omitted, the results data set specified by the &_cstResultsDS will be written to.
_cstLogLevel	Identifies the level of error reporting. Valid values (Info, Warning, Error, Fatal Error). Default=Info.

SAS® CLINICAL STANDARDS TOOLKIT AND ODM – VALIDATING ODM

What does 'validating' an ODM XML file really mean? We can think of 3 different levels of validation:

- Is the XML file well-formed?
- Is the XML file valid according to an XML schema?
- Does the content of the XML file conform to the specification rules?

We will take a look at these 3 levels of validation.

Well-formed XML

Well-formedness rules specify constraints such as "Every start-tag must have a matching end-tag", and "Attribute values must be quoted". Every XML document, without exception, must be well-formed. This means it must adhere to a number of rules, including the following:

- Proper element nesting is strictly enforced, and every open tag must be matched by a corresponding close tag.
- Elements may nest but may not overlap.
- There must be exactly one root element.
- Attribute values must be quoted.
- An element may not have two attributes with the same name.
- Comments and processing instructions may not appear inside tags.
- No unescaped < or & signs may occur in the character data of an element or attribute.

This is not an exhaustive list. There are many, many ways a document can be malformed [3].

Whether the error is small or large, likely or unlikely, an XML parser reading a document is required to report it. It may or may not report multiple well-formedness errors it detects in the document. However, the parser is not allowed to try to fix the document and make a best-faith effort of providing what it thinks the author really meant. It can't fill in missing quotes around attribute values, insert an omitted end-tag, or ignore the comment that's inside a start-tag. The parser is required to return an error. The SAS Clinical Standards Toolkit will report errors when trying to import an XML document that is not well-formed.

As an example, the following XML fragment is not well-formed, since the open tag is not matched by the close tag:

```
<ItemDef OID="ItemDef.OID.DM.SEX" Name="SEX" DataType="string" Length="2"
         SASFieldName="SEX" Comment="Sex of the subject. ">
    <CodeListRef CodeListOID="CodeLists.OID.SEX"/>
</Itemdef>
```

When validating this ODM file with the **odm_xmlvalidate** macro, the SAS Clinical Standards Toolkit reports the following in the Results SAS data set:

VIEWTABLE: Results.Read_results

	resultid	srcdata	message	resultseverity
32	ODM0003	XML VALIDATION	(Line 443/Column 15) The element type "ItemDef" must be terminated by the matching end-tag "</ItemDef>".	Error
33	ODM0003	XML TRANSFORMER	The element type "ItemDef" must be terminated by the matching end-tag "</ItemDef>".	Error

A subsequent call to the **odm_read** macro would not import this malformed ODM XML file, and the SAS Clinical Standards Toolkit would report in the SAS Log:

```
MPRINT(ODM_READ):   * set logging to INFO;
MPRINT(ODM_READ):   prefs.callvoidmethod('setLogLevelString','INFO');
MPRINT(ODM_READ):   dcl javaobj
transformer("com/sas/ptc/transform/xml/StandardXMLImporter",
prefs);
MPRINT(ODM_READ):   transformer.exceptiondescribe(1);
MPRINT(ODM_READ):   transformer.callvoidmethod('exec');
MPRINT(ODM_READ):   * check the return values here and get results path;
MPRINT(ODM_READ):   transformer.delete();
MPRINT(ODM_READ):   prefs.delete();
MPRINT(ODM_READ):   run;

ERROR:  'The element type "ItemDef" must be terminated by the matching end-tag
"</ItemDef>".'
ERROR:  'com.sun.org.apache.xml.internal.utils.WrappedRuntimeException: The element type
"ItemDef" must be terminated by the matching end-tag "</ItemDef>".'
```

Validating ODM – XML Schema Validation

XML schema validation of an ODM XML file involves verification that the XML file is valid structurally and syntactically according to the XML schema for the standard describing the ODM file. In our case this is the ODM 1.3.0 XML schema. Every XML file that validates according to an XML schema is well-formed, but not every well-formed XML file is valid.

The SAS Clinical Standards Toolkit includes a call to the **odm_xmlvalidate** macro immediately following the call to the **odm_write** macro as the last step of the create_odmxml.sas sample driver program. If you customize the ODM XML file after it is generated, then this macro can be used to validate the changes. The SAS Clinical Standards Toolkit also includes a call to the **odm_xmlvalidate** macro immediately before the call to the odm_read macro in the create_sasodm_fromxml.sas sample driver program.
As an example, the following XML fragment is well-formed, but not valid since the element should have been <ItemDef> instead of <Itemdef> for the Item with Name="SEX":

```
<ItemDef OID="ItemDef.OID.DM.RFSTDTC" Name="RFSTDTC" DataType="string" Length="64"
         SASFieldName="RFSTDTC"
         Comment="Reference Start Date/time for the subject in ISO 8601 character format.
Usually equivalent to date/time when subject was first exposed to study treatment. Required
for all randomized subjects; will be null for all subjects who did not meet the milestone the
date requires, such as screen failures or unassigned subjects."/>
<Itemdef OID="ItemDef.OID.DM.SEX" Name="SEX" DataType="string" Length="2"
         SASFieldName="SEX"
         Comment="Sex of the subject.">
  <CodeListRef CodeListOID="CodeLists.OID.SEX"/>
</Itemdef>
<ItemDef OID="ItemDef.OID.DM.SITEID" Name="SITEID" DataType="string" Length="40"
         SASFieldName="SITEID"
         Comment="Unique identifier for a site within a study."/>
```

When validating this ODM file with the **odm_xmlvalidate** macro, the SAS Clinical Standards Toolkit reports the following in the Results SAS data set:

VIEWTABLE: Results.Read_results

	resultid	srcdata	message	resultseverity
32	ODM0003	XML VALIDATION	(Line 441/Column 136) cvc-complex-type.2.4.a: Invalid content was found starting with element 'Itemdef'. One of '{"http://www.cdisc.org/ns/odm/v1.3":ItemDef, "http://www.cdisc.org/ns/odm/v1.3":CodeList, "http://www.cdisc.org/ns/odm/v1.3":ImputationMethod, "http://www.cdisc.org/ns/odm/v1.3":Presentation, "http://www.cdisc.org/ns/odm/v1.3":ConditionDef, "http://www.cdisc.org/ns/odm/v1.3":MethodDef}' is expected.	Error
33	ODM0002	XML TRANSFORMER	Document validation failed	Warning
34	ODM0011	ODM_XMLVALIDATE	Errors were reported in the generation of the ODM file.	Error

Since the ODM XML file is well-formed, the **odm_read** macro will still import this ODM file, ignoring the <Itemdef> element with Name="SEX".

VIEWTABLE: Srcdata.Itemdefs

	OID	Name	DataType	Length	SignificantDigits	SASFieldName
94	ItemDef.OID.DM.RACE	RACE	string	40	.	RACE
95	ItemDef.OID.DM.RFENDTC	RFENDTC	string	64	.	RFENDTC
96	ItemDef.OID.DM.RFSTDTC	RFSTDTC	string	64	.	RFSTDTC
97	ItemDef.OID.DM.SITEID	SITEID	string	40	.	SITEID
98	ItemDef.OID.DM.STUDYID	STUDYID	string	40	.	STUDYID

This example shows that the **odm_read** macro will attempt to import an invalid ODM XML file. However, importing an invalid ODM XML file may result in an incomplete import.

Validating ODM – Validation of the SAS® Data Set Representation

Validation of the SAS Representation of the ODM XML file relies on the definition of a master set of validation checks that are specific to the table and column metadata that define a set of data, and checks that are specific to the data itself. In the SAS Clinical Standards Toolkit, CDISC ODM validation uses the same types of metadata and the same workflow process that is common to validation of all data standards (e.g. SDTM, AdaM and CRT-DDS). SAS provides a set of validation checks for CDISC ODM that are designed to verify the metadata definitions and values of the default 66 data sets that comprise the SAS representation of the ODM model. These checks were created by SAS and are based on the CDISC ODM specification [2]. Metadata about each check is provided in the Validation Master data set in the folder <global standards library directory>/standards/cdisc-odm-1.3.0-1.4/validation/control. The **odm_validate** macro controls the validation workflow for ODM. As each check is processed from the run-time validation check data set, the check determines the source of the table and column metadata to use. The *reference_tables* and *reference_columns* data sets contain the metadata for the default 66 data sets that comprise the SAS representation for CDISC ODM. Unless you make customizations or run-time modifications, the source metadata *source_tables* and *source_columns* data sets contain the same content as the reference metadata *reference_tables* and *reference_columns* data sets. If all 66 ODM tables contribute information to the ODM XML file, then the validation process can run directly against the reference tables and columns data sets. In this case, the USESOURCEMETADATA data flag in the validation check data set needs to be set to N. However, you can elect to run validation against a subset of the 66 tables. In this case, a *source_tables* data set that contains the subset needs to be created from the *reference_tables* data set. And, a corresponding *source_columns* data set needs to be created from the *reference_columns* data set. The run-time validation check data set can contain all of the checks, and USESOURCEMETADATA can be left set to Y, which is the default value.

Table 2 enumerates the types of checks to be done on CRT-DDS data. Each check type is assumed to operate on data that exists in a source column within a source dataset. A check type may reference one or more parameters that will be used to validate the source column data. These parameters may be character strings, or may be a representation of some column other than the source column, against which the source column data must be compared. All character comparisons are case sensitive. Character data is assumed to have been 'trimmed' of any leading or trailing whitespace.

Table 2 CDISC ODM Validation Checks

Check Type	Category	Description
Unique in dataset	Structural	No two values for the source column can be equivalent within the same source dataset
Required character value	Data	The 'trimmed' (whitespace removed) value of the character data must consist of one or more characters.
Required numeric value	Data	The numeric value of the column cannot be 'missing'.
Enumeration (s0,s1, …)	Data	If character data exists, its value must match one of the given enumerated character strings. All string comparisons are case-sensitive.
Foreign key (targetColumn)	Structural	Each existing value in this column must have an equivalent value in the given target column.
Foreign key required (targetColumn)	Structural	A value is required for this column in every row, and each value must have an equivalent value in the given target column. This check is the equivalent of running the 'Required character value' check, and failing if that check fails. If 'required character value' passes, the 'foreign key()' check is run.
Character format: language	Data	The character data must consist of 1-8 alphabetical characters of either case, followed optionally by a hyphen (-) character and any sequence of 1-8 alphabetical characters of either case, or numeric digits, after that hyphen. For example, 'e' is a legal value, as is 'en-us' and 'english' and 'english-d842'. Illegal values include '1en', 'mumblespeak' and 'en_us'. The hyphen-character sequence can be repeated any number of times, also making a value such as 'english-mumbly-growly-47' a legal value. Regular expression: "[a-zA-Z]{1,8}(-[a-zA-Z0-9]{1,8})*"
Character format: fileName	Data	The character data must not contain any characters other than upper- and lower-case letters of the alphabet, numeric digits, the underscore (_) character or the period (.) character. Regular expression: `[A-Za-z0-9_.]+`
Character format: sasFormat	Data	The first character must be either a lower- or upper-case letter, an underscore (_), or the dollar sign ($). Any subsequent character must be either an upper- or lowercase letter, a numeric digit, the underscore (_), or a period (.). Regular expression: `[A-Za-z $][A-Za-z0-9 .]*`
Character format: sasName	Data	The first character must be either a lower- or upper-case letter, or an underscore (_). Any subsequent character must be either an upper- or lowercase letter, a numeric digit, or the underscore (_). Regular expression: `[A-Za-z ][A-Za-z0-9 ]*`
Unique across datasets (targetcolumn0, …)	Structural	No value in this column can be equal to any value in any of the given dataset column(s)
Primary key	Data	Must satisfy the 'Unique in dataset' check type as well as the 'Required character value' check type.
Must Have Corresponding Value (targetColumn)	Structural	For each distinct value in this column, there must be at least one equivalent value in the supplied target column.
No Duplicates Per Unique Value (targetColumn)	Structural	For each distinct value in the target column, each value in the source column must be unique. That is, the same value cannot appear more than once in the source column for each distinct value in the target column.

Let's have a look at an example.

```
<ItemGroupDef OID="ItemGroupDefs.OID.DM" Name="DM" Repeating="No"
              IsReferenceData="N" SASDatasetName="DM" Domain="DM"
              Purpose="Tabulation">
  <ItemRef ItemOID="ItemDef.DM.AGEU" Mandatory="No" OrderNumber="11" Role="RecordQualifier"/>
  <ItemRef ItemOID="ItemDef.DM.AGEU" Mandatory="No" OrderNumber="12" Role="VariableQualifier"/>
…
</ItemGroupDef>
…
<ItemDef OID=="ItemDef.DM.AGEU" Name="AGEU" DataType="string" Length="10"
         SASFieldName="AGEU" Comment="Units associated with AGE. ">
   <CodeListRef CodeListOID="CodeLists.OID.AGEU"/>
</ItemDef>

<CodeList OID="CodeLists.OID.AGEUNIT" Name="AGEU" DataType="string"
          SASFormatName="AGEU">
  <EnumeratedItem CodedValue="YEARS" />
</CodeList>
```

Although this ODM file is valid according to the XML schema, there are several issues:

- There is a duplicate "ItemDef.OID.DM.AGEU" ItemRef element
- The CodeList referenced (CodeLists.OID.AGU) does not exist
- The SASFormatName should start with a "$" character

When running the **odm_validate** macro on this ODM file, several issues are reported in the Results SAS data set:

VIEWTABLE: Results.Validation_results

	resultid	checkid	srcdata	message	resultseverity	resultflag	_cst_rc	actual
41	ODM0100	ODM0100	SRCDATA.ITEMGROUPDEFITEMREFS	No two values for the source column can be equivalent within the same source data set	Error	1	0	keys=FK_ITEMGROUPDEFS ITEMOID
201	ODM0110	ODM0110	SRCDATA.ITEMDEFS (SRCDATA.CODELISTS)	The foreign key OID does not have a corresponding value in the target data set SRCDATA.ITEMDEFS	Error	1	0	CODELISTREF=CodeLists.OID.AGEU
323	ODM0217	ODM0217	SRCDATA.CODELISTS	A CodeList contains a SASFormatName value that does not have the correct format: CodeList datatype is text or string and the FormatName does not start with $.	Error	1	0	DATATYPE=string,SASFORMATNAME=AGEU

SAS® CLINICAL STANDARDS TOOLKIT AND ODM – CLINICAL DATA IN ODM

Starting in ODM 1.3.0, there are two forms of the ItemData element -- the element used by the ODM for transmitting clinical data item values. These are the untyped and typed forms.
An example of a typed ItemData element and an untyped ItemData element:

```
<ItemDataFloat ItemOID="ID.VS.VSSTRESN">76.1</ItemDataFloat>
<ItemData ItemOID="ID.AE.AETERM" Value="HEADACHE" />
```

Both of these data values will be stored in the Value variable in the ItemData SAS data set. In the case of typed data the ItemDataType variable in the ItemData SAS data set will have the data type (e.g. Float). In the case of untyped data the ItemDataType variable in the ItemData SAS data set will be null.
Normally, Typed and untyped data transmission should not be mixed within a single ODM file. However, in the example provided by the Toolkit both transmission types are part of the same example for demonstration purposes.

CONCLUSION

With the production releases of SAS Clinical Standards Toolkit 1.4, the CDISC ODM reference standard will support reading and representing in SAS a complete ODM XML file, building an ODM XML file, and validating the structure and content of the SAS representation of an ODM XML file. In addition, it will validate the structural integrity of the ODM XML file.

REFERENCES

1. SAS Institute, Inc.: SAS® Clinical Standards Toolkit 1.4: User Guide. Cary, NC, USA: SAS Institute Inc. 2011. Available from: http://support.sas.com/rnd/base/cdisc/cst/index.html
2. CDISC Operational Data Model (ODM), Version 1.3.0, December, 2006 (http://www.cdisc.org/odm)
3. Eliotte Rusty Harold & W. Scott Means, 2004, XML in a Nutshell, *A Desktop Quick Reference* (O'Reilly and Associates)

CONTACT INFORMATION

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

Lex Jansen
SAS Institute Inc.
SAS Campus Drive
Cary, North Carolina 27513 USA
Phone: 919-531-9860
Email: lex.jansen@sas.com

Appendix 1: ODM 1.3.0 SAS data sets

Below is the complete listing of the SAS data sets, with member columns for each, which will comprise the ODM 1.3.0 data in the SAS Clinical Standards Toolkit v.1.4.
No dataset has more than one variable that acts as they key/index for that table. The names of key variables are prepended with two asterisks (**). Some tables do not have any key at all. Foreign key variables' names are prepended with two carat characters (^^) and reference (in [brackets]) the name of the dataset for which it is a foreign key. Required fields (fields for which, in any observation for that dataset, a non-nil, and non-whitespace-only value must be supplied) are marked with an 'X' between brackets: [X].
Per the standard, which is very flexible, only the DefineDocument dataset, containing valid values for the FileOID and FileType variables, are needed in order to create a minimal, but valid ODM-compliant XML document.
All table and column names are case-sensitive, and must be specified exactly as shown.

In the SAS implementation of the relational data model, the keys were extended to define a unique record in every SAS data set. For example, a unique record in the EnumeratedItems SAS data set is defined by the variables FK_CODELISTS and CODEDVALUE. These SAS data set keys can be found in the SAS *reference_tables* SAS data set, which contains the metadata for the tables that define the SAS representation of the ODM 1.3.0 standard.

SAS Dataset Name	Variable Name	SAS Data Type	Length (if char)
ODM			
	**FileOID	character	64 (oid)
	Archival	character	3
	AsOfDateTime	character	32
	CreationDateTime	character	32
	Description	character	2000 (text)
	FileType	character	13
	Granularity	character	15
	Id	character	64 (oid)
	ODMVersion	character	2000 (text)
	Originator	character	2000 (text)
	PriorFileOID	character	64 (oid)
	SourceSystem	character	2000 (text)
	SourceSystemVersion	character	2000 (text)
Study			
	**OID	character	64 (oid)
	StudyName	character	128 (name)
	StudyDescription	character	2000 (text)
	ProtocolName	character	128 (name)
	^^FK_ODM [ODM]	character	64 (oid)
MeasurementUnits			
	**OID	character	64 (oid)
	Name	character	128 (name)
	^^FK_Study [Study]	character	64 (oid)
MUTranslatedText			
	TranslatedText	character	2000 (text)
	lang	character	128 (name)
	^^FK_MeasurementUnits [MeasurementUnits]	character	64 (oid)
MetaDataVersion			
	**OID	character	64 (oid)
	Name	character	128 (name)
	Description	character	2000 (text)
	IncludedOID	character	64 (oid)
	IncludedStudyOID	character	64 (oid)
	^^FK_Study [Study]	character	64 (oid)
ProtocolEventRefs			
	Mandatory	character	3
	OrderNumber	numeric	8
	^^StudyEventOID [StudyEventDefs]	character	64 (oid)
	^^FK_MetaDataVersion [MetaDataVersion]	character	64 (oid)
	^^CollectionExceptionConditionOID [ConditionDefs]	character	64 (oid)
ProtocolTranslatedText			
	TranslatedText	character	2000 (text)
	lang	character	17
	^^FK_MetaDataVersion [MetaDataVersion]	character	64 (oid)
StudyEventDefs			
	**OID	character	64 (oid)

SAS Dataset Name	Variable Name	SAS Data Type	Length (if char)
	Category	character	2000 (text)
	Name	character	128 (name)
	Repeating	character	3
	Type	character	11
	^^FK_MetaDataVersion [MetaDataVersion]	character	64 (oid)
StudyEventDefTranslatedText			
	TranslatedText	character	2000 (text)
	lang	character	17
	^^FK_StudyEventDefs [StudyEventDefs]	character	64 (oid)
StudyEventFormRefs			
	^^FormOID [FormDefs]	character	64 (oid)
	Mandatory	character	3
	OrderNumber	numeric	8
	^^FK_StudyEventDefs [StudyEventDefs]	character	64 (oid)
	^^CollectionExceptionConditionOID [ConditionDefs]	character	64 (oid)
FormDefs			
	**OID	character	64 (oid)
	Name	character	128 (name)
	Repeating	character	3
	^^FK_MetaDataVersion [MetaDataVersion]	character	64 (oid)
FormDefTranslatedText			
	TranslatedText	character	2000 (text)
	lang	character	17
	^^FK_FormDefs [FormDefs]	character	64 (oid)
FormDefItemGroupRefs			
	^^ItemGroupOID [ItemGroupDefs]	character	64 (oid)
	Mandatory	character	3
	OrderNumber	numeric	8
	^^FK_FormDefs [FormDefs]	character	64 (oid)
	^^CollectionExceptionConditionOID [ConditionDefs]	character	64 (oid)
FormDefArchLayouts			
	**OID	character	64 (oid)
	PdfFileName	character	512 (path)
	^^PresentationOID [Presentation]	character	64 (oid)
	^^FK_FormDefs [FormDefs]	character	64 (oid)
ItemGroupDefs			
	**OID	character	64 (oid)
	Name	character	128 (name)
	Repeating	character	3
	IsReferenceData	character	3
	SASDatasetName	character	8
	Domain	character	2000 (text)
	Origin	character	2000 (text)
	Role	character	128 (name)
	Purpose	character	2000 (text)
	Comment	character	2000 (text)
	^^FK_MetaDataVersion [MetaDataVersion]	character	64 (oid)
ItemGroupDefTranslatedText			
	TranslatedText	character	2000 (text)
	lang	character	17
	^^FK_ItemGroupDefs [ItemGroupDefs]	character	64 (oid)
ItemGroupDefItemRefs			
	^^ItemOID [ItemDefs]	character	64 (oid)
	Mandatory	character	3
	OrderNumber	numeric	8
	KeySequence	numeric	8
	^^ImputationMethodOID [ImputationMethods]	character	64 (oid)
	^^MethodOID [MethodDefs]	character	64 (oid)
	Role	character	128 (name)
	^^RoleCodeListOID [CodeLists]	character	64 (oid)
	^^FK_ItemGroupDefs [ItemGroupDefs]	character	64 (oid)
	^^CollectionExceptionConditionOID [ConditionDefs]	character	64 (oid)
ItemGroupAliases			
	Context	character	2000 (text)
	Name	character	2000 (text)
	^^FK_ItemGroupDefs [ItemGroupDefs]	character	64 (oid)
ItemDefs			

SAS Dataset Name	Variable Name	SAS Data Type	Length (if char)
	**OID	character	64 (oid)
	Name	character	128 (name)
	DataType	character	18
	Length	numeric	8
	SignificantDigits	numeric	8
	SASFieldName	character	8
	SDSVarName	character	8
	Origin	character	2000 (text)
	Comment	character	2000 (text)
	^^CodeListRef [CodeLists]	character	64 (oid)
	^^FK_MetaDataVersion [MetaDataVersion]	character	64 (oid)
ItemDefTranslatedText			
	TranslatedText	character	2000 (text)
	lang	character	17
	^^FK_ItemDefs [ItemDefs]	character	64 (oid)
ItemQuestionTranslatedText			
	TranslatedText	character	2000 (text)
	lang	character	17
	^^FK_ItemDefs [ItemDefs]	character	64 (oid)
ItemQuestionExternal			
	Dictionary	character	2000 (text)
	Version	character	2000 (text)
	Code	character	2000 (text)
	^^FK_ItemDefs [ItemDefs]	character	64 (oid)
ItemMURefs			
	^^MeasurementUnitOID [MeasurementUnits]	character	64 (oid)
	^^FK_ItemDefs [ItemDefs]	character	64 (oid)
ItemRangeChecks			
	**OID	character	64 (oid)
	Comparator	character	5
	SoftHard	character	4
	^^MURefOID [MeasurementUnits]	character	64 (oid)
	^^FK_ItemDefs [ItemDefs]	character	64 (oid)
ItemRangeCheckValues			
	CheckValue	character	512 (value)
	^^FK_ItemRangeChecks [ItemRangeChecks]	character	64 (oid)
RCErrorTranslatedText			
	TranslatedText	character	2000 (text)
	lang	character	17
	^^FK_ItemRangeChecks [ItemRangeChecks]	character	64 (oid)
ItemRCFormalExpression			
	Context	character	2000 (text)
	Expression	character	2000 (text)
	^^FK_ItemRangeChecks [ItemRangeChecks]	character	64 (oid)
ItemRole			
	Name	character	2000 (text)
	^^FK_ItemDefs [ItemDefs]	character	64 (oid)
ItemAliases			
	Context	character	2000 (text)
	Name	character	2000 (text)
	^^FK_ItemDefs [ItemDefs]	character	64 (oid)
CodeLists			
	**OID	character	64 (oid)
	Name	character	128 (name)
	DataType	character	7
	SASFormatName	character	8
	^^FK_MetaDataVersion [MetaDataVersion]	character	64 (oid)
ExternalCodeLists			
	Dictionary	character	2000 (text)
	Version	character	2000 (text)
	ref	character	2000 (text)
	href	character	2000 (text)
	^^FK_CodeLists [CodeLists]	character	64 (oid)
EnumeratedItems			
	CodedValue	character	512 (value)
	Rank	numeric	8
	^^FK_CodeLists [CodeLists]	character	64 (oid)

SAS Dataset Name	Variable Name	SAS Data Type	Length (if char)
CodeListItems			
	##OID	character	64 (oid)
	CodedValue	character	512 (value)
	^^FK_CodeLists [CodeLists]	character	64 (oid)
	Rank	numeric	8
CLItemDecodeTranslatedText			
	TranslatedText	character	2000 (text)
	lang	character	17
	^^FK_CodeListItems [CodeListItems]	character	64 (oid)
ImputationMethods			
	**OID	character	64 (oid)
	method	character	2000 (text)
	^^FK_MetaDataVersion [MetaDataVersion]	character	64 (oid)
Presentation			
	**OID	character	64 (oid)
	presentation	character	2000 (text)
	lang	character	17
	^^FK_MetaDataVersion [MetaDataVersion]	character	64 (oid)
MethodDefs			
	**OID	character	64 (oid)
	Name	character	128 (name)
	Type	character	11
	^^FK_MetaDataVersion [MetaDataVersion]	character	64 (oid)
MethodDefTranslatedText			
	TranslatedText	character	2000 (text)
	lang	character	17
	^^FK_MethodDefs [MethodDefs]	character	64 (oid)
MethodDefFormalExpression			
	Context	character	2000 (text)
	Expression	character	2000 (text)
	^^FK_MethodDefs [MethodDefs]	character	64 (oid)
ConditionDefs			
	**OID	character	64 (oid)
	Name	character	128 (name)
	^^FK_MetaDataVersion [MetaDataVersion]	character	64 (oid)
ConditionDefTranslatedText			
	TranslatedText	character	2000 (text)
	lang	character	17
	^^FK_ConditionDefs [ConditionDefs]	character	64 (oid)
ConditionDefFormalExpression			
	Context	character	2000 (text)
	Expression	character	2000 (text)
	^^FK_ConditionDefs [ConditionDefs]	character	64 (oid)
ClinicalData			
	##OID	character	64 (oid)
	^^StudyOID [Study]	character	64 (oid)
	^^MetaDataVersionOID [MetaDataVersion]	character	64 (oid)
	^^FK_ODM [ODM]	character	64 (oid)
SubjectData			
	##OID	character	64 (oid)
	SubjectKey	character	2000 (text)
	TransactionType	character	7
	^^InvestigatorRefOID [User]	character	64 (oid)
	^^SiteRefOID [Location]	character	64 (oid)
	^^FK_ClinicalData [ClinicalData]	character	64 (oid)
StudyEventData			
	##OID	character	64 (oid)
	^^StudyEventOID [StudyEventDefs]	character	64 (oid)
	StudyEventRepeatKey	character	2000 (text)
	TransactionType	character	7
	^^FK_SubjectData [SubjectData]	character	64 (oid)
FormData			
	##OID	character	64 (oid)
	^^FormOID [FormDefs]	character	64 (oid)
	FormRepeatKey	character	2000 (text)
	TransactionType	character	7
	^^ArchiveLayoutRefOID [FormDefArchLayouts]	character	64 (oid)

SAS Dataset Name	Variable Name	SAS Data Type	Length (if char)
	^^FK_StudyEventData [StudyEventData]	character	64 (oid)
ItemGroupData			
	##OID	character	64 (oid)
	^^ItemGroupOID [ItemGroupDefs]	character	64 (oid)
	ItemGroupRepeatKey	character	2000 (text)
	TransactionType	character	7
	^^FK_FormData [FormData]	character	64 (oid)
	^^FK_ReferenceData [ReferenceData]	character	64 (oid)
ItemData			
	##OID	character	64 (oid)
	^^ItemOID [ItemDefs]	character	64 (oid)
	TransactionType	character	7
	TransactionOrder	numeric	8
	Value	character	2000 (text)
	IsNull	character	3
	ItemDataType	character	18
	^^FK_ItemGroupData [ItemGroupData]	character	64 (oid)
	AuditRecordID [AuditRecords]	character	64 (oid)
	SignatureID [Signatures]	character	64 (oid)
	AnnotationID [Annotations]	character	64 (oid)
	MeasurementUnitOID [MeasurementUnits]	character	64 (oid)
ReferenceData			
	##GeneratedID	character	64 (oid)
	^^StudyOID [Study]	character	64 (oid)
	^^MetaDataVersionOID [MetaDataVersion]	character	64 (oid)
	^^FK_ODM [ODM]	character	64 (oid)
AdminData			
	##GeneratedID	character	64 (oid)
	StudyOID	character	64 (oid)
	^^FK_ODM [ODM]	character	64 (oid)
User			
	**OID	character	64 (oid)
	UserType	character	12 (enum)
	LoginName	character	2000 (text)
	DisplayName	character	2000 (text)
	FullName	character	2000 (text)
	FirstName	character	2000 (text)
	LastName	character	2000 (text)
	Organization	character	2000 (text)
	PictureImageType	character	2000 (text)
	PictureFileName	character	2000 (text)
	Pager	character	2000 (text)
	^^FK_AdminData [AdminData]	character	64 (oid)
UserAddress			
	##GeneratedID	character	64 (oid)
	City	character	2000 (text)
	StateProv	character	2000 (text)
	Country	character	2 (text)
	PostalCode	character	2000 (text)
	OtherText	character	2000 (text)
	^^FK_User [User]	character	64 (oid)
UserAddressStreetName			
	StreetName	character	2000 (text)
	^^FK_UserAddress [UserAddress]	character	64 (oid)
UserEmail			
	Email	character	2000 (text)
	^^FK_User [User]	character	64 (oid)
UserFax			
	Fax	character	2000 (text)
	^^FK_User [User]	character	64 (oid)
UserPhone			
	Phone	character	2000 (text)
	^^FK_User [User]	character	64 (oid)
UserLocationRef			
	^^LocationOID [Location]	character	64 (oid)
	^^FK_User [User]	character	64 (oid)
Location			

SAS Dataset Name	Variable Name	SAS Data Type	Length (if char)
	**OID	character	64 (oid)
	Name	character	2000 (text)
	LocationType	character	7 (enum)
	^^FK_AdminData [AdminData]	character	64 (oid)
LocationVersion			
	^^StudyOID [Study]	character	64 (oid)
	^^MetaDataVersionOID [MetaDataVersion]	character	64 (oid)
	EffectiveDate	character	10 (date)
	^^FK_Location [Location]	character	64 (oid)
SignatureDef			
	**OID	character	64 (oid)
	Methodology	character	10 (enum)
	Meaning	character	2000 (text)
	LegalReason	character	2000 (text)
	^^FK_AdminData [AdminData]	character	64 (oid)
AuditRecord			
	ID	character	2000 (text)
	EditPoint	character	14 (enum)
	UsedImputationMethod	character	3 (Yes/No)
	^^UserOID [User]	character	64 (oid)
	^^LocationOID [Location]	character	64 (oid)
	DateTimeStamp	character	25 (datetime)
	ReasonForChange	character	2000 (text)
	SourceID	character	2000 (text)
	ParentType	character	14 (datanode)
	ParentKey	character	64 (oid)
Signature			
	ID	character	2000 (text)
	^^UserOID [User]	character	64 (oid)
	^^LocationOID [Location]	character	64 (oid)
	^^SignatureDefOID [SignatureDef]	character	64 (oid)
	DateTimeStamp	character	25 (datetime)
	ParentType	character	14 (datanode)
	ParentKey	character	64 (oid)
Annotation			
	##GeneratedID	character	64 (oid)
	ID	character	2000 (text)
	SeqNum	numeric	8 (integer)
	TransactionType	character	7 (enum)
	CommentSponsorOrSite	character	7 (enum)
	Comment	character	2000 (text)
	ParentType	character	14 (datanode)
	ParentKey	character	64 (oid)
AnnotationFlag			
	FlagValue	character	2000 (text)
	^^FlagValueCodeListOID [CodeLists]	character	64 (oid)
	FlagType	character	128 (name)
	^^FlagTypeCodeListOID [CodeLists]	character	64 (oid)
	^^FK_Annotation [Annotation]	character	64 (oid)
Association			
	##GeneratedID	character	64 (oid)
	^^StudyOID [Study]	character	64 (oid)
	^^MetaDataVersionOID [MetaDataVersion]	character	64 (oid)
	^^FK_ODM [ODM]	character	64 (oid)
KeySet			
	**OID	character	64 (oid)
	^^StudyOID [Study]	character	64 (oid)
	SubjectKey	character	2000 (text)
	^^StudyEventOID [StudyEventDefs]	character	64 (oid)
	StudyEventRepeatKey	character	2000 (text)
	^^FormOID [FormDefs]	character	64 (oid)
	FormRepeatKey	character	2000 (text)

SAS Dataset Name	Variable Name	SAS Data Type	Length (if char)
	^^ItemGroupOID [ItemGroupDefs]	character	64 (oid)
	ItemGroupRepeatKey	character	2000 (text)
	^^ItemOID [ItemDefs]	character	64 (oid)
	^^FK_Association [Association]	character	64 (oid)