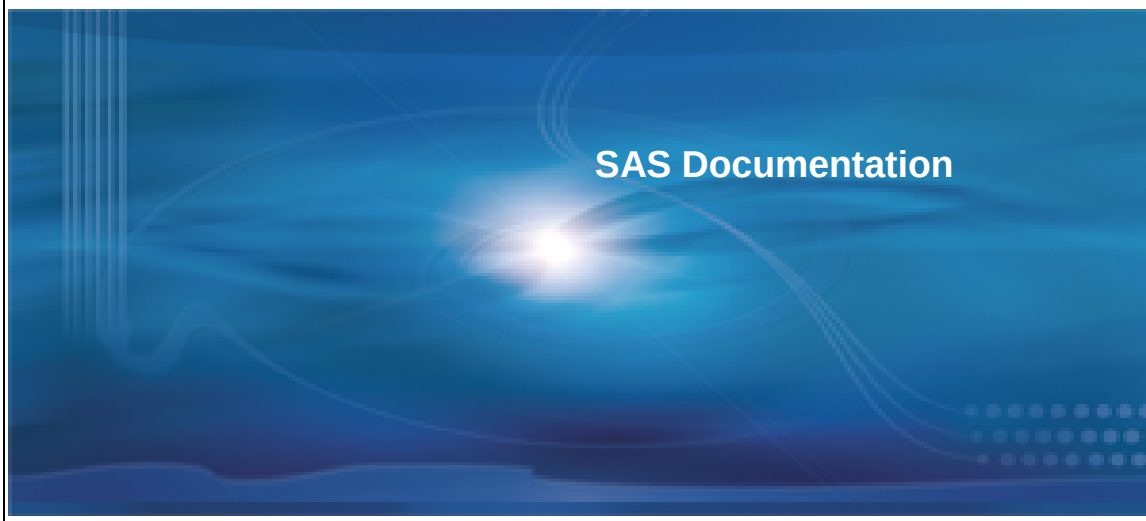
SAS® Macros to support Dataset-XML v1.0.0



SAS® Macros to support Dataset-XML v1.0.0

Prepared date 2Jul2014

Introduction

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.

On November 5, 2012, the FDA held a meeting entitled “Regulatory New Drug Review: Solutions for Study Data Exchange Standards”, the purpose of which was to solicit input regarding the advantages and disadvantages of current and emerging open, consensus-based standards for the exchange of regulated study data. Dataset-XML [1] was presented as an alternative for consideration.

Dataset-XML defines a standard format for transporting tabular data in XML between any two entities based on the CDISC ODM [2]. That is, in addition to supporting the transport of data sets as part of a submission to the FDA, it may also be used to facilitate other data interchange use cases. For example, the Dataset-XML data format can be used by a CRO to transmit SDTM or ADaM data sets to a sponsor organization. Dataset-XML supports SDTM, ADaM, and SEND CDISC data sets but can also be used to exchange any other type of tabular data set.

Dataset-XML and Define-XML

Dataset-XML defines a standard format for transporting tabular data set data in XML. The metadata for a data set contained within a Dataset-XML document must be specified using the Define-XML standard. Each Dataset-XML file contains data for a single data set, but a single Define-XML file describes all the data sets included in the folder. Both Define-XML v1.0 [3] and Define-XML v2.0 [4] are supported for use with Dataset-XML.

Requirements

The macros that are part of this download package require Base SAS. They have been tested on SAS 9.4 (TS1M1) on Windows x64 and Linux x64.

The macros have been successfully used on SAS 9.3 M2 (Windows x64), SAS 9.2 (Windows XP), and SAS 9.3 on UNIX.

The validation of XML files against an XML schema uses PROC GROOVY, which has been available since SAS 9.3.

Availability

Download the macros as a ZIP file from the SAS® Clinical Standards Toolkit page:
http://support.sas.com/rnd/base/cdisc/cst/index.html

You are encouraged to download and use this new functionality. To help guide future development, post feedback on the SAS in Health Care Related Fields and Clinical Trials forum.

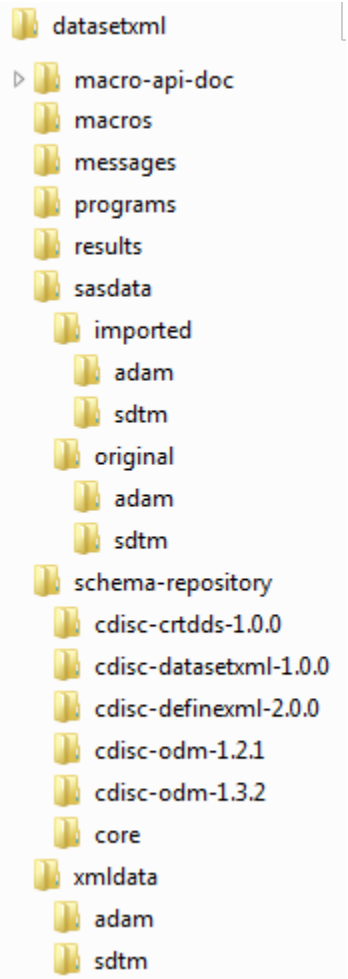
Note: The functionality of the macros will be added to the SAS Clinical Standards Toolkit 1.7.

Installation and Folder Structure

Open the ZIP file, and extract the files to a folder on your computer.

Note: The sample programs in this document assume the folder structure shown in Display 1.

Display 1 *Dataset-XML Folder Structure*



Note: The sample programs use the DatasetXMLRoot macro variable, which specifies the root of the Dataset-XML folder structure. If you unzip to a root location other than /datasetxml, you must change the value of DatasetXMLRoot. Here is an example of the default specification for DatasetXMLRoot:

```
************************************************;
* Location of the Dataset-XML root               *;
************************************************;
%let DatasetXMLRoot=/datasetxml;
```

```
************************************************;
* Set Root paths for input and output            *;
************************************************;
%let studyRootPath=/datasetxml;
%let studyOutputPath=/datasetxml;
```

The ZIP file provides SAS data sets to demonstrate the macros. These SAS data sets were created on Windows x64. On other operating systems, these SAS data sets might need to be extracted from a SAS CPORT transport file. You can perform the extraction by running this program:

Dataset-XML Root/programs/cimport.sas

This program extracts **cported.stc** files that are located in folders that contain SAS data sets.

The folder structure contains these subfolders:

Table 1 *Folder Structure in the ZIP File*

macro-api-doc	The macro API documentation for the macros described in this document. The index file is: *Dataset-XML Root*/macro-api-doc/index.html
macros	The macros that are used by the sample programs.
messages	The data set that contains a list of codes and associated message text that are used by the macros.
programs	The sample programs that demonstrate the functionality of the macros. - Creating Dataset-XML files from SAS data sets - Creating SAS data sets from Dataset-XML files - Validating Dataset-XML files against an XML schema - Comparing original SAS data sets with SAS data sets created from Dataset-XML files All sample programs contain examples for SDTM and ADaM.
results	The data sets created by the sample programs.
sasdata	The original SAS data sets used to create the Dataset-XML files and the SAS data sets that are imported from the Dataset-XML files. Sample data sets are provided both for SDTM 3.1.2 and ADaM 2.1.
schema-repository	The XML schema files that Dataset-XML and Define-XML files are validated against. These XML schema files are published by CDISC.
xmldata	The Dataset-XML files that are created from SAS data sets and the Define-XML files that define the metadata.

Creating Dataset-XML Files from SAS Data Sets

This sample program creates Dataset-XML files from a library of SAS data sets:

Dataset-XML Root/programs/create_datasetxml_standalone.sas

After initialization, the program runs the datasetxml_write macro:

```
linage sdtmdata  "&studyRootPath/sasdata/original/sdtm";
filename defxml "&studyOutputPath/xmldata/sdtm/define.xml";
libname dataxml  "&studyOutputPath/xmldata/sdtm";

%datasetxml_write(
  _cstSourceLibrary=sdtmdata,
  _cstOutputLibrary=dataxml,
  _cstSourceMetadataDefineFileRef=defxml,
  _cstCheckLengths=Y,
  _cstIndent=N,
  _cstZip=Y,
  _cstDeleteAfterZip=N
  );
```

The Define-XML file that describes the SAS data sets must contain metadata information about all SAS data sets and all variables to be converted. The Dataset-XML files by themselves do not have any information about the SAS data sets (name and label) or the SAS variables (name, label, data type, length, and display format). When the Dataset-XML file is converted back to SAS data sets, this information must be provided by the Define-XML file.

Display 2 and Display 3 illustrate a Dataset-XML file and a Define-XML file.

Display 2 *Dataset-XML Fragment*

```
<?xml version="1.0" encoding="UTF-8"?>
<ODM xmlns="http://www.cdisc.org/ns/odm/v1.3"
  xmlns:data="http://www.cdisc.org/ns/Dataset-XML/v1.0"
  ODMVersion="1.3.2" FileType="Snapshot" FileOID="cdisc01.AE"
  PriorFileOID="www.cdisc.org.Studycdisc01-Define-XML_2.0.0"
  CreationDateTime="2014-06-23T13:18:18"
  data:DatasetXMLVersion="1.0.0">
  <ClinicalData StudyOID="cdisc01"
    MetaDataVersionOID="MDV.CDISC01.SDTMIG.3.1.2.SDTM.1.2">
    <ItemGroupData ItemGroupOID="IG.AE" data:ItemGroupDataSeq="1">
      ...
      <ItemData ItemOID="IT.AE.AETERM" Value="AGITATED"/>
```

Display 3 *Define-XML Fragment*

```
<ODM ... >

  <Study OID="cdisc01">
    ...
    <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="SDTM-IG"
      def:StandardVersion="3.1.2">
  ...
      <ItemGroupDef OID="IG.AE"
        Domain="AE" Name="AE" Repeating="Yes" IsReferenceData="No"
        SASDatasetName="AE" Purpose="Tabulation"
        def:Structure="One record per adverse event per subject"
        def:Class="EVENTS" def:ArchiveLocationID="LF.AE">
        ...
        <ItemRef ItemOID="IT.AE.AETERM" OrderNumber="6" Mandatory="Yes"/>
  ...
      <ItemDef OID="IT.AE.AETERM" Name="AETERM" DataType="text" Length="25"
        SASFieldName="AETERM">
```

In a Dataset-XML file, the ClinicalData attributes StudyOID and MetaDataVersionOID must be the same value as the corresponding OID attributes in the define.xml document. The ItemGroupOID value must be the same value as the corresponding ItemGroup OID attribute.

In the Dataset-XML file, all ItemOID attributes in the ItemData elements must have values identical to the values of the corresponding ItemOID attributes in the ItemRef elements that are child elements of the corresponding ItemGroupDef element in the define.xml document.

It would be an error to try to extract from the Dataset-XML file the SAS data set name from an ItemGroup object identifier (`ItemGroupOID="IG.AE"`) or to extract the variable name from an object identifier (`ItemOID="IT.AE.AETERM"`). There is no requirement concerning the values of the identifiers.

SAS tables and columns are matched to @SASDatasetName (or, if this value is not specified, @Name) and @SASFieldName (or, if this value is not specified, @Name). SASDatasetName and SASFieldName are optional but @Name is always available.

In case the ItemGroup or ItemDef is not found, the XML is generated with the following pattern for @ItemGroupOID and @ItemOID:

> ItemGroupOID = ”IG.<table>”
> ItemOID = “IT.<table>.<column>”

Although ItemGroupOID and ItemOID are generated for missing ItemGroups or missing ItemDefs, it is important to realize that this may lead to problems later when converting Dataset-XML files to SAS data sets.

Warnings are written to the SAS log file and the write_results data set in the results folder. Here is an example:

```
WARNING: [CSTLOGMESSAGE.DATASETXML_WRITE] Columns not found in metadata:
ADAE.AEDECOD ADAE.AETERM
WARNING: [CSTLOGMESSAGE.DATASETXML_WRITE] Missing ItemData/@ItemOID for
column=AEDECOD has been set to IT.ADAE.AEDECOD
WARNING: [CSTLOGMESSAGE.DATASETXML_WRITE] Missing ItemData/@ItemOID for
column=AETERM has been set to IT.ADAE.AETERM
```

70	DATA0097	1	DATASETXML_WRITE	Metadata used from C:\datasetxml\xmldata\adam\define_adam.xml	Info
71	DATA0098	2	DATASETXML_WRITE	Columns not found in metadata: ADAE.AEDECOD ADAE.AETERM	Warning
72	DATA0098	3	DATASETXML_WRITE	Missing ItemData/@ItemOID for these columns will be generated as IT.<TABLE>.<COLUMN>	Warning
73	DATA0097	4	DATASETXML_WRITE	ADAMDATA.ADAE converted to C:\datasetxml\xmldata\adam\adae.xml in 0.39 seconds (106 records).	Info
74	DATA0097	5	DATASETXML_WRITE	Zip file C:\datasetxml\xmldata\adam\adae.zip was created	Info

The @IsReferenceData attribute in the Define-XML file determines whether the data set is considered ReferenceData or ClinicalData:

```
<ReferenceData StudyOID="cdisc01"
  MetaDataVersionOID="MDV.CDISC01.SDTMIG.3.1.2.SDTM.1.2">
```

```
    <ItemGroupData ItemGroupOID="IG.TE" data:ItemGroupDataSeq="1">
      <ItemData ItemOID="IT.STUDYID" Value="CDISC01"/>
      <ItemData ItemOID="IT.TE.DOMAIN" Value="TE"/>
      <ItemData ItemOID="IT.TE.ETCD" Value="EOS"/>
      <ItemData ItemOID="IT.TE.ELEMENT" Value="End of Study"/>
      <ItemData ItemOID="IT.TE.TESTRL" Value="Study Termination"/>
      <ItemData ItemOID="IT.TE.TEDUR" Value="P1D"/>
    </ItemGroupData>

  <ClinicalData StudyOID="cdisc01"
    MetaDataVersionOID="MDV.CDISC01.SDTMIG.3.1.2.SDTM.1.2">
    <ItemGroupData ItemGroupOID="IG.AE" data:ItemGroupDataSeq="1">
      <ItemData ItemOID="IT.STUDYID" Value="CDISC01"/>
      <ItemData ItemOID="IT.AE.DOMAIN" Value="AE"/>
      <ItemData ItemOID="IT.USUBJID" Value="CDISC01.100008"/>
      <ItemData ItemOID="IT.AE.AESEQ" Value="1"/>
      <ItemData ItemOID="IT.AE.AESPID" Value="1"/>
      <ItemData ItemOID="IT.AE.AETERM" Value="AGITATED"/>
```

Although not essential for the creation of Dataset-XML files, setting the _cstCheckLengths macro parameter to “Y” enables the macro to determine whether the lengths defined in the metadata are long enough for character data. Warnings are written to the SAS log file and the write_results data set in the results folder. Here is an example:

```
WARNING: [CSTLOGMESSAGE.DATASETXML_WRITE] Length too short:
__ItemGroupOID=IG.ADAE __ItemOID=IT.ADAE.AETERM Length=20 _valueLength=24
value=HEARTBURN-LIKE DYSPEPSIA
WARNING: [CSTLOGMESSAGE.DATASETXML_WRITE] Length too short:
__ItemGroupOID=IG.ADAE __ItemOID=IT.ADAE.AETERM Length=20 _valueLength=25
value=ACID REFLUX (OESOPHAGEAL)
WARNING: [CSTLOGMESSAGE.DATASETXML_WRITE] Length too short:
__ItemGroupOID=IG.ADAE __ItemOID=IT.ADAE.AEDECOD Length=20 _valueLength=32
value=Gastrooesophageal reflux disease
WARNING: [CSTLOGMESSAGE.DATASETXML_WRITE] Length too short:
__ItemGroupOID=IG.ADAE __ItemOID=IT.ADAE.AETERM Length=20 _valueLength=25
value=ACID REFLUX (OESOPHAGEAL)
```

8	DATA0097	1	DATASETXML_WRITE	Metadata used from C:\SASPlaypen\TK1.7\US7405_CreateDatasetXML\sourcexml_adam\define_adam.xml	Info
9	DATA0098	2	DATASETXML_WRITE	Check Log for potential length issues: ADAE.AEDECOD ADAE.AETERM	Warning
10	DATA0097	3	DATASETXML_WRITE	SRCDATA.ADAE converted to C:\SASPlaypen\TK1.7\US7405_CreateDatasetXML\sourcexml_adam\adae.xml	Info
11	DATA0097	4	DATASETXML_WRITE	Zip file C:\SASPlaypen\TK1.7\US7405_CreateDatasetXML\sourcexml_adam\adae.zip was created	Info

This check is important to avoid data truncation problems when importing the Dataset-XML files into SAS data set with the datasetxml-read macro.

The datasetxml_write macro also checks that numeric variables in ADaM data sets that represent Date/Time information have a DisplayFormat defined in the Define-XML file.

Creating SAS Data Sets from Dataset-XML files

This sample program creates SAS data sets from Dataset-XML files:

Dataset-XML Root/programs/create_sas_from_datasetxml_standalone.sas

After initialization, the program runs the datasetxml_read macro:

```
libname dataxml  "&studyRootPath/xmldata/sdtm";
libname sdtmdata  "&studyRootPath/sasdata/imported/sdtm";
filename defxml  "&studyOutputPath/xmldata/sdtm/define.xml";

%datasetxml_read(
  _cstSourceDatasetXMLLibrary=dataxml,
  _cstOutputLibrary=sdtmdata,
  _cstSourceMetadataDefineFileRef=defxml,
  _cstDatetimeLength=64,
  _cstAttachFormats=Y,
  _cstNumObsWrite=10000
  );
```

The Define-XML file that describes the Dataset-XML files must contain metadata information about all Dataset-XML files and all variables to be converted to SAS data sets. The Dataset-XML files by themselves do not have any information about the SAS data sets (name and label) or the SAS variables (name, label, data type, length, and display format).

Character variables that represent Date/Time related information in ADaM or SDTM data conform to the ISO 8601 standard and do not have a length specified in the Define-XML file. The _cstDateTimeLength macro parameter specifies the length to use for these variables when they are converted to SAS data sets. If the lengths of character variables are too short to hold the data, warnings are written to the SAS log file and the read_results data set in the results folder. Here is an example of length issues:

```
WARNING: [CSTLOGMESSAGE.DATASETXML_READ] TRUNCATION occurred: Length=20 too
short for ItemGroupDataSeq=12 IT.ADAE.AETERM value=HEARTBURN-LIKE DYSPEPSIA
(length=
24)
WARNING: [CSTLOGMESSAGE.DATASETXML_READ] TRUNCATION occurred: Length=20 too
short for ItemGroupDataSeq=25 IT.ADAE.AETERM value=HEARTBURN-LIKE DYSPEPSIA
(length=
24)
WARNING: [CSTLOGMESSAGE.DATASETXML_READ] TRUNCATION occurred: Length=20 too
short for ItemGroupDataSeq=28 IT.ADAE.AETERM value=ACID REFLUX (OESOPHAGEAL)
WARNING: [CSTLOGMESSAGE.DATASETXML_READ] TRUNCATION occurred: Length=20 too
short for ItemGroupDataSeq=28 IT.ADAE.AEDECOD value=Gastrooesophageal reflux
disease
(length=32)
```

71	DATA0097	2	DATASETXML_READ	Data read from C:\datasetxml\xmldata\adam\adae.xml	Info
72	DATA0098	3	DATASETXML_READ	Please check the LOG. There were data truncation issues for table ADAE	Warning
73	DATA0097	4	DATASETXML_READ	Data set adamdata.ADAE created in 0.92 seconds (106 records)	Info

Inconsistencies between the Dataset-XML file and the Define-XML file, which can lead to issues with matching data to metadata, are written to the SAS log file and the read_results data set in the results folder. Here is an example:

```
WARNING: [CSTLOGMESSAGE.DATASETXML_READ] Items not found in metadata:
IT.ADAE.AEDECOD IT.ADAE.AETERM
```

70	DATA0097	1	DATASETXML_READ	Metadata used from C:\datasetxml\xmldata\adam\define_adam.xml	Info
71	DATA0097	2	DATASETXML_READ	Data read from C:\datasetxml\xmldata\adam\adae.xml	Info
72	DATA0098	3	DATASETXML_READ	Items not found in metadata: IT.ADAE.AEDECOD IT.ADAE.AETERM	Warning
73	DATA0097	4	DATASETXML_READ	Data set adamdata.ADAE created in 0.96 seconds (106 records)	Info

In this example, the ADAE data set is created without the AETERM and AEDECOD variables, as shown in this PROC COMPARE output:

```
Dataset                   Created          Modified  NVar    NObs  Label

DATABASE.ADAE  23JUN14:09:01:29  23JUN14:09:01:29    61     106  Adverse Event
Analysis Dataset
DATACOMP.ADAE  02JUL14:12:58:14  02JUL14:12:58:14    59     106  Adverse Event
Analysis Dataset


Variables Summary

Number of Variables in Common: 59.
Number of Variables in DATABASE.ADAE but not in DATACOMP.ADAE: 2.


Listing of Variables in DATABASE.ADAE but not in DATACOMP.ADAE

Variable  Type  Length  Label

AETERM    Char     200  Reported Term for the Adverse Event
AEDECOD   Char     200  Dictionary-Derived Term
```

37	DATA0097	1	CSTUTILCOMPAREDATASETS	Base library: database (C:\datasetxml\sasdata\original\adam)	Info	0	0	
38	DATA0097	2	CSTUTILCOMPAREDATASETS	Comp library: datacomp (C:\datasetxml\sasdata\imported\adam)	Info	0	0	
39	DATA0099	1	CSTUTILCOMPAREDATASETS	Comparing database.adae and datacomp.adae - Differences	Error	1024	1024	BASEVAR

The datasetxml_read macro can require significant memory and disc space. To help fine tune system performance, the macro _cstNumObsWrite specifies the maximum number of observations to write per loop in the final DATA step. The default value is 10,000. In case of memory issues, decrease the number.

Note: Detailed information about fine tuning system performance is beyond the scope of this document.

Validating Dataset-XML Files Against an XML Schema

This sample program validates Dataset-XML files and the Define-XML file against the XML schema as it was published by CDISC:

Dataset-XML Root/programs/validate_datasetxml.sas

Here is an example call:

```
%xml_validate(
  xsdFile=/datasetxml/schema-repository/cdisc-datasetxml-1.0.0/dataset1-0-0.xsd,
  xmlFolder=/datasetxml/xmldata/sdtm,
  whereClause=%nrstr(where index(upcase(xmlfile), "DEFINE") eq 0)
  );

%xml_validate(
  xsdFile=/datasetxml/schema-repository/cdisc-definexml-2.0.0/define2-0-0.xsd,
  xmlFolder=/datasetxml/xmldata/sdtm,
  whereClause=%nrstr(where index(upcase(xmlfile), "DEFINE") ge 1)
  );
```

The whereClause macro parameter limits validation to specific files or excludes specific files.

Here is an example of the success messages that are written to the SAS log file:

```
NOTE: [CSTLOGMESSAGE] Validating /datasetxml//xmldata/adam/adae.xml against
/datasetxml/schema-repository/cdisc-datasetxml-1.0.0/dataset1-0-0.xsd
NOTE: [CSTLOGMESSAGE] Validating /datasetxml//xmldata/adam/adqs.xml against
/datasetxml/schema-repository/cdisc-datasetxml-1.0.0/dataset1-0-0.xsd
NOTE: [CSTLOGMESSAGE] Validating /datasetxml//xmldata/adam/adsl.xml against
/datasetxml/schema-repository/cdisc-datasetxml-1.0.0/dataset1-0-0.xsd
NOTE: [CSTLOGMESSAGE] Validating /datasetxml//xmldata/adam/adtte.xml against
/datasetxml/schema-repository/cdisc-datasetxml-1.0.0/dataset1-0-0.xsd
```

Here is an example of the validation errors that are written to the SAS log file:

```
NOTE: [CSTLOGMESSAGE] Validating /datasetxml/xmldata/sdtm/define.xml against
/datasetxml/schema-repository/cdisc-definexml-2.0.0/define2-0-0.xsd
ERROR: [CSTLOGMESSAGE]line 2292, col 101 : cvc-enumeration-valid: Value
'Numeric' is not facet-valid with respect to enumeration '[integer, float,
date,
datetime, time, text, string, double, URI, boolean, hexBinary, base64Binary,
hexFloat, base64Float, partialDate, partialTime, partialDatetime,
durationDatetime, intervalDatetime, incompleteDatetime, incompleteDate,
incompleteTime]'. It must be a value from the enumeration.
ERROR: [CSTLOGMESSAGE]line 2292, col 101 : cvc-attribute.3: The value 'Numeric'
of attribute 'DataType' on element 'ItemDef' is not valid with respect to its
type, 'DataType'.
ERROR: [CSTLOGMESSAGE]line 2298, col 97 : cvc-datatype-valid.1.2.1: '' is not a
valid value for 'integer'.
ERROR: [CSTLOGMESSAGE]line 2298, col 97 : cvc-attribute.3: The value '' of
attribute 'Length' on element 'ItemDef' is not valid with respect to its type,
'positiveInteger'.
```

Comparing Original SAS Data Sets to SAS Data Sets Created from Dataset-XML Files

This sample program compares the original SAS data sets with the SAS data sets that were created from the Dataset-XML files:

Dataset-XML Root/programs/compare_datasetxml.sas

After initialization, the program runs the cstutilcomparedatasets macro:

```
libname database "&studyRootPath/sasdata/original/sdtm";
```

```
libname datacomp "&studyRootPath/sasdata/imported/sdtm";

%cstutilcomparedatasets(
  _cstLibBase=database,
  _cstLibComp=datacomp,
  _cstCompareLevel=16,
  _cstCompOptions=%str(criterion=0.00000000000001),
  _cstCompDetail=Y
  );
```

For every SAS data set that is compared, the macro reports the error code as returned by PROC COMPARE. Here are the error codes:

Bit	Condition	Code	Hex	Description
1	DSLABEL	1	0001X	Data set labels differ
2	DSTYPE	2	0002X	Data set types differ
3	INFORMAT	4	0004X	Variable has different informat
4	FORMAT	8	0008X	Variable has different format
5	LENGTH	16	0010X	Variable has different length
6	LABEL	32	0020X	Variable has different label
7	BASEOBS	64	0040X	Base data set has observation not in comparison
8	COMPOBS	128	0080X	Comparison data set has observation not in base
9	BASEBY	256	0100X	Base data set has BY group not in comparison
10	COMPBY	512	0200X	Comparison data set has BY group not in base
11	BASEVAR	1024	0400X	Base data set has variable not in comparison
12	COMPVAR	2048	0800X	Comparison data set has variable not in base
13	VALUE	4096	1000X	A value comparison was unequal
14	TYPE	8192	2000X	Conflicting variable types
15	BYVAR	16384	4000X	BY variables do not match
16	ERROR	32768	8000X	Fatal error: comparison not done

For example, `code=40 (8+32)` indicates that a format and a label were different.
This message is written to the SAS log file:

```
WARNING: [CSTLOGMESSAGE.CSTUTILCOMPAREDATASETS] Comparing srcdata.adqs
and trgdata.adqs  - Differences: FORMAT/LABEL (SysInfo=40)
```

When converting SAS data sets to Dataset-XML and then converting back to SAS data sets, expect these differences:

- Date- and time-related columns do not have a length defined in the Define-XML metadata. Specify a length to assign in the macro (for example, `_cstdatetimeLength=64`). The length is used to create the date- and time related

columns but can be different from the original lengths.

- SAS numeric variables are created with a length of 8 to avoid loss of precision, even when the original length or the length specified in the Define-XML file is less than 8.
- Character variables (DataType="text") that do not have a length specified in the Define-XML file are created with a length of 200.
- Small differences in precision can be expected around the machine precision for numeric variables that represent real numbers.
- Character data that contains leading spaces or trailing spaces loses the leading and trailing spaces.

By specifying PROC COMPARE options with the _cstCompOptions macro parameter, you can specify that the comparison be less precise (for example, _cstCompOptions=%str(criterion=0.00000000000001)). Lesser precision prevents differences close to machine precision from being reported as errors.

References

1. CDISC Dataset-XML Specification Version 1.0, April 22, 2014 (http://www.cdisc.org/dataset-xml)
2. CDISC Operational Data Model (ODM), Version 1.3.2, February 5, 2013 (http://www.cdisc.org/odm)
3. Case Report Tabulation Data Definition Specification (define.xml), Version 1.0.0, February 9, 2005 (http://www.cdisc.org/define-xml)
4. CDISC Define-XML Specification, Version 2.0, March 5, 2013 (http://www.cdisc.org/define-xml)