EFFECTIVELY PROCESSING XML USING SAS

CHEVELL PARKER
TECHNICAL SUPPORT ANALYST, SAS INSTITUTE INC.
Overview

- XML basics and Libname engine introduction
- Effectively reading XML files
- Introduction to the XML Mapper
- Writing XML files using SAS
- Common problems processing XML files
Processing XML Files Using SAS

- Writing files
- XML Mapper
- Common Errors
- Reading XML
What is XML?

• XML is a set of rules used for defining & modeling structures
• XML is extensible & customizable
  ▪ Its greatest strength
  ▪ Its greatest weakness
XML Basics- Well Formed Files

- Document has a single root element
- Elements nest properly
- No tag omission (close what you open)
- Attributes must be quoted
  - Special characters &lt; &gt; and & must always be escaped
- XML is case sensitive
Anatomy of an XML file

<?xml version="1.0"?>
<workorder priority="high" datedue="09/30/2001">
  <submitter>
    <name first="Jennifer" last="Kyrnin" />
    <email>html.guide@about.com</email>
    <account number="11001100" />
  </submitter>
  <project title="update aa051198.htm article">
    <url>http://webdesign.com/aa051198.htm</url>
    <description>new article</description>
  </project>
</workorder>
# Libname Engine Introduction

<table>
<thead>
<tr>
<th>Release</th>
<th>Engine</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAS 8.1</td>
<td>XML</td>
<td>Export production</td>
</tr>
<tr>
<td>SAS 8.2</td>
<td>XML</td>
<td>Ability to read added with hot fix</td>
</tr>
<tr>
<td>SAS 9.1</td>
<td>XML</td>
<td></td>
</tr>
<tr>
<td>SAS 9.2</td>
<td>XML</td>
<td>XML92</td>
</tr>
<tr>
<td>SAS 9.3</td>
<td>9.4</td>
<td>XML</td>
</tr>
</tbody>
</table>
Libname Engine Introduction-Processing XML

- Provides the ability to read and write XML files
- Requires that XML files be well formed for reading
- Reads and writes generic XML files by default
Libname Engine Introduction-Extended Functionality of the XMLV2 Engine

- Wildcards can be used to read all files in a directory
- Allows hierarchical files to be read by dynamically generating Map files
- Namespaces are supported beginning with SAS 9.3
- Provides the ability to use XMLMap files for export as well as import
Effectively Reading XML Files

```xml
<?xml version="1.0" ?>
<TABLE> root node
  <STUDENTS> repeating element instance
    <ID> 0755 </ID> begin reading row 1
    <NAME> Brad Martin </NAME>
    <ADDRESS> 1611 Glengreen </ADDRESS>
    <CITY> Huntsville </CITY>
    <STATE> Texas </STATE>
  </STUDENTS>

  <STUDENTS> repeating element instance
    <ID> 1522 </ID> begin reading row 2
    <NAME> Zac Harvell </NAME>
    <ADDRESS> 11900 Glenda </ADDRESS>
    <CITY> Houston </CITY>
    <STATE> Texas </STATE>
  </STUDENTS>
</TABLE>
```
Effectively Reading XML Files

Libname test xmlv2 'C:\students.xml';
proc print data=test.students;
run;

<table>
<thead>
<tr>
<th>ID</th>
<th>NAME</th>
<th>ADDRESS</th>
<th>CITY</th>
<th>STATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0755</td>
<td>Brad Martin</td>
<td>1611 Glengreen</td>
<td>Huntsville</td>
<td>Texas</td>
</tr>
<tr>
<td>1522</td>
<td>Zac Harvell</td>
<td>11900 Glenda</td>
<td>Houston</td>
<td>Texas</td>
</tr>
</tbody>
</table>
Effectively Reading XML Files - Attributes Require an XMLMap

```xml
<?xml version="1.0"?>
<TABLE>
  <STUDENTS Dept="Math">
    <ID> 0755 </ID>
    <NAME> Brad Martin </NAME>
    <ADDRESS> 1611 Glengreen </ADDRESS>
    <CITY> Huntsville </CITY>
    <STATE> Texas </STATE>
  </STUDENTS>

  <STUDENTS Dept="CSC">
    <ID> 1522 </ID>
    <NAME> Zac Harvell </NAME>
    <ADDRESS> 11900 Glenda </ADDRESS>
    <CITY> Houston </CITY>
    <STATE> Texas </STATE>
  </STUDENTS>
</TABLE>
```
Effectively Reading XML Files-Attributes Require an XMLMap

Libname test xmlv2 'C:\students.xml';
proc print data=test.students;
run;

<table>
<thead>
<tr>
<th>ID</th>
<th>NAME</th>
<th>ADDRESS</th>
<th>CITY</th>
<th>STATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0755</td>
<td>Brad Martin</td>
<td>1611 Glengreen</td>
<td>Huntsville</td>
<td>Texas</td>
</tr>
<tr>
<td>1522</td>
<td>Zac Harvell</td>
<td>11900 Glenda</td>
<td>Houston</td>
<td>Texas</td>
</tr>
</tbody>
</table>
## Effectively Reading XML Files-Required Structure

Problem.XML

<table>
<thead>
<tr>
<th><code>&lt;xml version=&quot;1.0&quot; ?&gt;</code></th>
<th>Declaration statement</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;Table&gt;</code></td>
<td>SAS recognizes the first instance tag as the root enclosing element, which is the document container.</td>
</tr>
<tr>
<td><code>&lt;Students&gt;</code></td>
<td>Starting with the second-level instance tag, SAS begins to scan for columns.</td>
</tr>
<tr>
<td><code>&lt;Student&gt;</code></td>
<td>SAS expects to start reading data here and because there is no data an error occurs</td>
</tr>
<tr>
<td><code>&lt;ID&gt; 0755 &lt;/ID&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;NAME&gt; Brad Martin&lt;/NAME&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;ADDRESS&gt; 1611 Glengreen&lt;/ADDRESS&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;CITY&gt; Huntsville &lt;/CITY&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;STATE&gt; Texas &lt;/STATE&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;/Student&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;/Students&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;/Table&gt;</code></td>
<td></td>
</tr>
</tbody>
</table>
libname temp xmlv2 'c:\problem.xml';

proc copy in=temp out=work;
run;

ERROR: XML data is not in a format supported natively by the XML libname engine. Files of this type may require an XMLMap to be input properly.

NOTE: Statements not processed because of errors noted above.

NOTE: PROCEDURE COPY used (Total process time):
    real time     0.10 seconds
    cpu time      0.01 seconds
Effectively Reading XML Files-XMLMap Files

• An XML file that describes XML markup to the engine
• Allows meta data to be added such as the data type, formats and informats, lengths and more
• Element names can be modified making them legal for SAS
• XMLMap files can be created a variety of ways
• Maps function much like the industry standard XML schema
Effectively Reading XML Files - XMLMap Files and XPATH Syntax

<?xml version="1.0" ?>
<LIBRARY>
  <STUDENTS>
    <ID>0755</ID>
    <NAME>Brad Martin</NAME>
    <ADDRESS>1611 Glengreen</ADDRESS>
    <CITY>Huntsville</CITY>
    <STATE>Texas</STATE>
  </STUDENTS>
</LIBRARY>
The XML Mapper
The XML Mapper – Loading Files to Map
The XML Mapper – AUTOMAP Feature

Automap button

XML code:
```xml
<?xml version="1.0" ?>
<LIBRARY>
  <STUDENTS Dept="Math">
    <ID>0266</ID>
    <NAME>Brad Martin</NAME>
    <ADDRESS>1011 Glengreen</ADDRESS>
    <CITY>Huntsville</CITY>
    <STATE>Texas</STATE>
  </STUDENTS>

  <STUDENTS Dept="CSC">
    <ID>1622</ID>
    <NAME>Zac Harrell</NAME>
    <ADDRESS>11300 Glenda</ADDRESS>
    <CITY>Houston</CITY>
    <STATE>Texas</STATE>
  </STUDENTS>
</LIBRARY>
```
The XML Mapper – AUTOMAP Feature

Tables created
XML Mapper-AUTOMAP

- Generates one or more tables depending on file structure
- Creates ordinal columns which acts as a keys by default
- Character values use length of largest value
- Can generate map files from a Schema or XML data files
<?xml version="1.0" ?>
<PHARMACY>
  (1)
  <PERSON> (2) 
    <NAME>Brad Martin</NAME>
    <STREET>11900 Glenda Court</STREET>
    <CITY>Austin</CITY>
    <PRESCRIPTION> (3) 
      <DRUG>Tetracycline</DRUG> (4)
      <DRUG>Lomotil</DRUG>
    </PRESCRIPTION>
  </PERSON>
  <PERSON>
    <NAME>Jim Spano</NAME>
    <STREET>1611 Glengreen</STREET>
    <CITY>Austin</CITY>
    <PRESCRIPTION>
      <DRUG>Nexium</DRUG>
    </PRESCRIPTION>
  </PERSON>
</PHARMACY>
XML Mapper- AUTOMAP and Table Generation
XML Mapper Icons

Elements

[ ]
Empty element

[ ]
Element with attributes

[ ]
Element with child elements

[ ]
Element with data

[ ]
Element with both attributes and child elements

[ ]
Element with both data and attributes

[ ]
Element with both data and child elements

[ ]
Element with data, attributes, and child elements
The XML Mapper – Creating Custom Map Files

- Create the correct table boundary on the file
- Avoiding truncated records
- Create custom ordinal by modifying path
- Changing formats and meta data
- Other advanced features
The XML Mapper – Custom Maps and Combining a Tables

```sas
filename person 'C:\person.xml';
filename SXLEMAP 'C:\person.map';
libname person xmlv2 xmlmap=SXLEMAP ;

data prescription;
  merge person.PRESCRIPTION person.DRUG;
  by prescription_ordinal;
run;

data person;
  merge person.person prescription;
  by person_ordinal;
run;

proc print;
run;
```
### The XML Mapper – Custom Maps and Combining a Tables

<table>
<thead>
<tr>
<th>PHARMACY_PERSON_</th>
<th>PRESCRIPTION_DRUG_</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obs</td>
<td>ORDINAL</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>
The XML Mapper – Modifying Column Names and Formats
Dynamically Generating XMLMap files from SAS
Beginning with Version 9.3M2

filename data 'c:\Wexample.xml';
filename map 'c:\Wexample.map';

libname data xmlv2 xmlmap=map automap=replace;

proc copy in=data out=work;
run;
Exporting XML Files using the XMLV2 Engine

libname temp xmlv2 'c:\example.xml';

data temp.class;
  set sashelp.class;
run;

<?xml version="1.0" encoding="WINDOWS-1252"?>
  <TABLE>
    <CLASS>
      <Name>Joyce</Name>
      <Sex>F</Sex>
      <Age>11</Age>
      <Height>51.3</Height>
      <Weight>50.5</Weight>
    </CLASS>
    <CLASS>
      <Name>Thomas</Name>
      <Sex>M</Sex>
      <Age>11</Age>
      <Height>57.5</Height>
      <Weight>85</Weight>
    </CLASS>
    <CLASS>
      <Name>James</Name>
      <Sex>M</Sex>
      <Age>12</Age>
      <Height>57.3</Height>
      <Weight>83</Weight>
    </CLASS>
    <CLASS>
      <Name>Jane</Name>
      <Sex>F</Sex>
      <Age>12</Age>
      <Height>59.8</Height>
      <Weight>84.5</Weight>
  </TABLE>
Exporting XML Files using the XMLV2 Engine

libname temp xmlv2 'c:\example.xml' xmltype=oracle;

data temp.class;
    set sashelp.class;
run;

Oracle

<?xml version="1.0" encoding="WINDOWS-1252"?>
- <ROWSET>
  - <ROW>
    <Name> Joyce </Name>
    <Sex> F </Sex>
    <Age> 11 </Age>
    <Height> 51.3 </Height>
    <Weight> 50.5 </Weight>
  </ROW>
  - <ROW>
    <Name> Thomas </Name>
    <Sex> M </Sex>
    <Age> 11 </Age>
    <Height> 57.5 </Height>
    <Weight> 85 </Weight>
  </ROW>
  - <ROW>
    <Name> James </Name>
    <Sex> M </Sex>
    <Age> 12 </Age>
    <Height> 57.3 </Height>
    <Weight> 83 </Weight>
  </ROW>
  - <ROW>
    <Name> Jane </Name>
    <Sex> F </Sex>
    <Age> 12 </Age>
    <Height> 59.8 </Height>
    <Weight> 84.5 </Weight>
Exporting XML Files using the XMLV2 Engine-Tagsets

- Custom tagsets can be generated to control the layout of export XML file
- Other shipped tagsets can be used to control the tagging structure such as how missing values are displayed
- The SASXMOG tagset is the default tagset used by the XML and XMLV2 engines
Common Questions – Where can I get a copy of the XML mapper?

- Install it from the demos and download site located at support.sas.com website or SAS Media
- It’s a stand alone Java application so it needs to follow the JRE guidelines
- If SAS versions prior to SAS 9.3 should use SAS 9.2 version of the Mapper
Common Questions – I am getting an Out of Memory Error using the XML mapper

- Create a map from a Schema (XSD) file rather than the XML data file
- Map a simple representation from the file rather than the entire file
- Increase the heap space for the JRE using the -XMX and -XMS options
Common Questions – Why am I getting Validate Errors Loading XML Files in the Mapper?

Modify parsing validation level
Common Questions – Why am I getting Transcode Errors Using the XMLV2 Engine

- Characters in the file cannot be represented correctly with the current session encoding
- They were ignored with the XML engine and not the XMLV2 engine
- Future option may be implemented to restore the behavior of the XML Engine
Common Questions – Why am I getting the File does not exist Error?

```sas
filename xmldata 'c:\example.xml';
libname xmldata xmlv2;

data new;
  set xmldata.example;
run;
```

Verify that this is a valid table by the XMLv2 Engine

```
NOTE: Libref XMLDATA was successfully assigned as follows:
  Engine:  XMLV2
  Physical Name:  c:\example.xml

data new;
set xmldata.class;
ERROR: File XMLDATA.class.DATa does not exist.
run;
```

NOTE: The SAS System stopped processing this step because of errors.
WARNING: The data set WORK.NEW may be incomplete. When this step was stopped there were 0 observations and 0 variables.
WARNING: Data set WORK.NEW was not replaced because this step was stopped.
NOTE: Data statement used (Total process time):
  real time  0.00 seconds
  cpu time   0.01 seconds
```
Resources

Base Engine
http://support.sas.com/rnd/base/xmlengine/index.html

XML Engine Tip Sheet
Contact

Email: Chevell.Parker@sas.com