Paper 099-2008

SAS® XML Mapper to the Rescue

Carol Martell, UNC Highway Safety Research Center, Chapel Hill, NC

ABSTRACT

Have you ever been given XML data without the requisite Document Type Definition (DTD) or XML Schema describing the data? Certain XML formats are supported in SAS through the `XMLTYPE=` option in the XML `LIBNAME` statement. If your data conforms to one of the supported formats, then you're in luck. If not, you have a problem. The tags surrounding each data item can seduce an accomplished programmer into writing a DATA step to read the data. While this is certainly possible, it isn't necessary. The SAS XML Mapper is a separate Java application from SAS that can, among other things, create a custom SAS XMLMap to allow you to read the unsupported XML. The SAS XMLMap is used by the XML engine to interpret an XML document. The XML libname engine becomes your friend again when you use the `XMLMAP=` option to point to your new XMLMap file.

INTRODUCTION

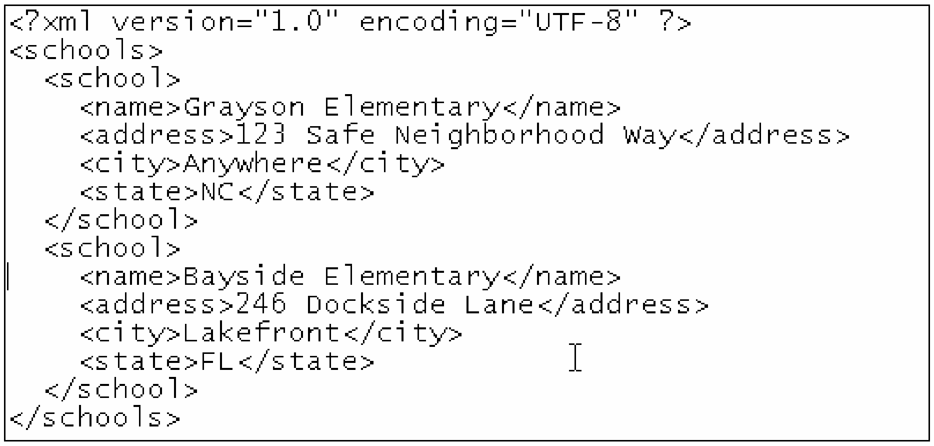
XML is a general-purpose markup language that enables the sharing of data in a non-proprietary format. SAS provides tools to support the use of this data format; one tool is the SAS XML Mapper. It is a Java-based drag-and-drop interface that generates the syntax for a SAS XMLMap, describing the tables and variables contained within the XML file. A SAS XMLMap is required to import XML data when the structure is either not generic, was not generated by MS Access (v9.0), or by (v9.1.3): CDISC Operational Data Model, ORACLE, OIMDBM/EXPORT, or HTML.

SIMPLE, GENERIC……?

Those words keep appearing wherever references are made to XML that doesn't require an XMLMap. How can you tell if your XML file qualifies as simple and generic? You can look at the data in a text editor and make a pretty good guess. After an opening tag announcing that XML follows, paired markup tags surround data items:

```
<SOMEDATA> A DATA ITEM </SOMEDATA>
```

The tags are identical except for the forward slash in the closing tag. Other tag pairs can be nested between opening and closing tags. Although nesting can continue to many levels, the amount of nesting is limited in simple, generic XML. The SAS documentation suggests that one think of the first tag level as the library and the first nested tag as the table level, with each pair representing a row in the table. Each variable is nested between the row tags. Any further nesting disqualifies the XML from being considered simple, generic XML and you'll need an XMLMap. To illustrate, in **Figure 1,** *name*, *address*, *city* and *state* are variables in the *school* table in the library *schools*.



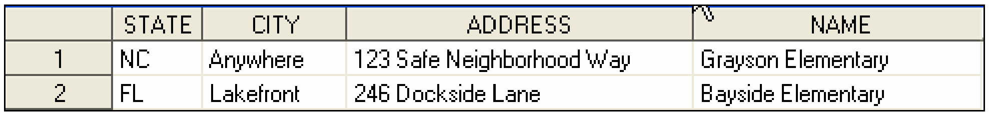
```
<?xml version="1.0" encoding="UTF-8" ?>
<schools>
  <school>
    <name>Grayson Elementary</name>
    <address>123 Safe Neighborhood Way</address>
    <city>Anywhere</city>
    <state>NC</state>
  </school>
  <school>
    <name>Bayside Elementary</name>
    <address>246 Dockside Lane</address>
    <city>Lakefront</city>
    <state>FL</state>
  </school>
</schools>
```

Figure 1 – Simple, Generic XML

The following code imports the **Figure 1** XML data, copying it to the work table *schools*, shown in **Figure 2**.

```
LIBNAME MYDATA XML 'C:\...\MYXML.XML';

DATA SCHOOLS; SET MYDATA.SCHOOL; RUN;
```

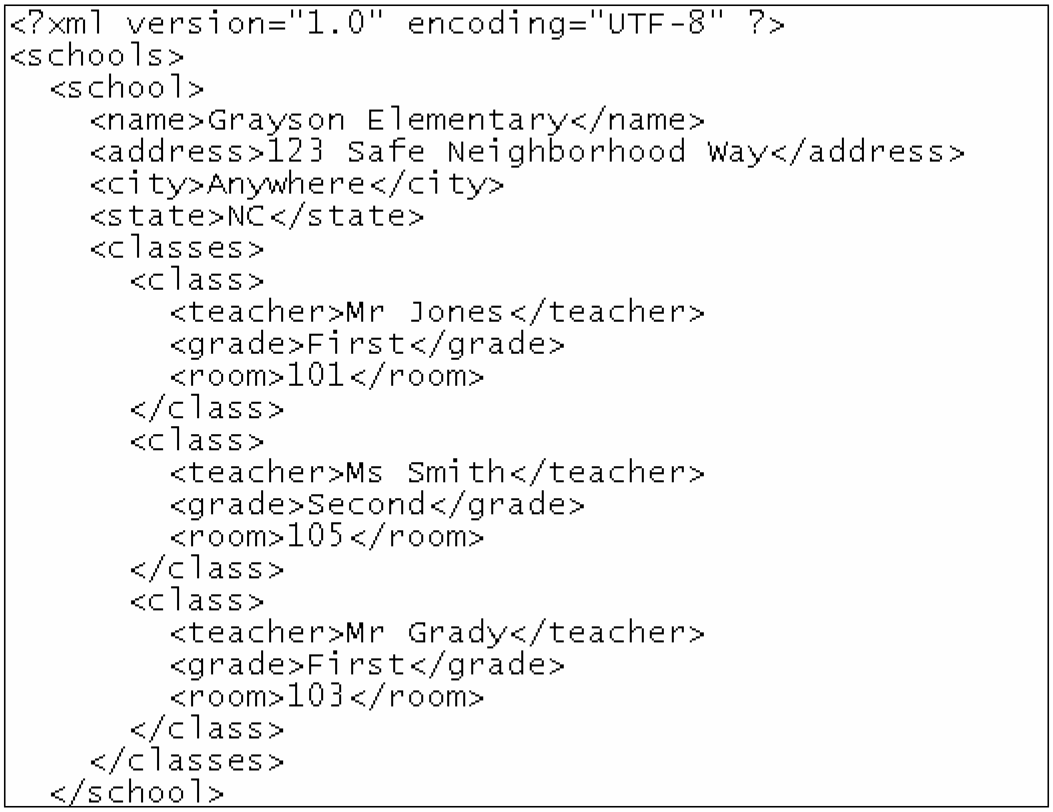


	STATE	CITY	ADDRESS	NAME
1	NC	Anywhere	123 Safe Neighborhood Way	Grayson Elementary
2	FL	Lakefront	246 Dockside Lane	Bayside Elementary

Figure 2 – Figure 1 XML data imported and copied to SAS table

NOT SO SIMPLE OR GENERIC

If further nesting levels occur in the XML, SAS cannot read the data. In **Figure 3** we have added information about classes to the XML file. In addition to the original information, each school now has a nested `<classes>` pair, containing rows describing each class.

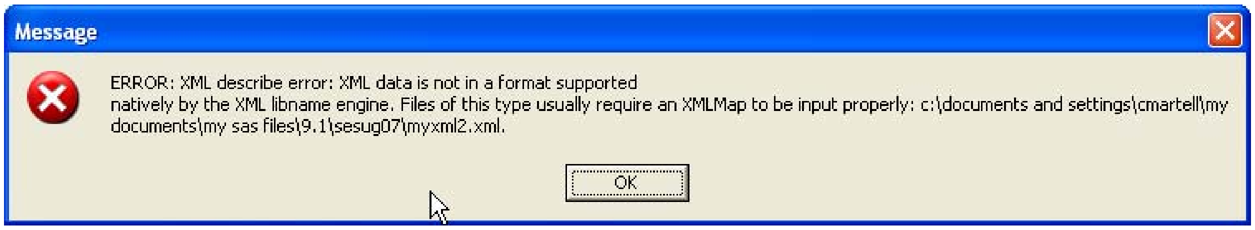


```
<?xml version="1.0" encoding="UTF-8" ?>
<schools>
  <school>
    <name>Grayson Elementary</name>
    <address>123 Safe Neighborhood Way</address>
    <city>Anywhere</city>
    <state>NC</state>
    <classes>
      <class>
        <teacher>Mr Jones</teacher>
        <grade>First</grade>
        <room>101</room>
      </class>
      <class>
        <teacher>Ms Smith</teacher>
        <grade>Second</grade>
        <room>105</room>
      </class>
      <class>
        <teacher>Mr Grady</teacher>
        <grade>First</grade>
        <room>103</room>
      </class>
    </classes>
  </school>
```

Figure 3 - School XML data with nested class information

We submit the following `LIBNAME` statement in SAS and receive the error shown in **Figure 4**.

```
LIBNAME MYDATA XML 'C:\...\MYXML2.XML';
```

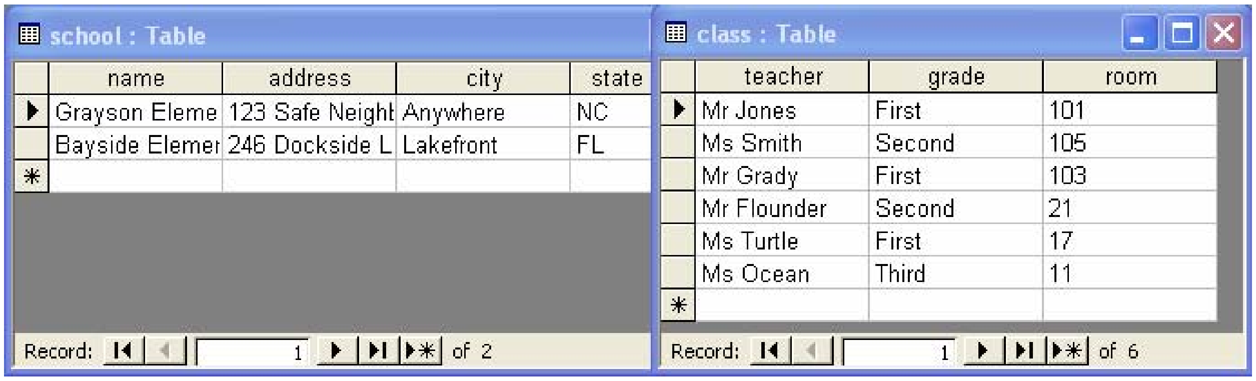


```
ERROR: XML describe error: XML data is not in a format supported
natively by the XML libname engine. Files of this type usually require an XMLMap to be input properly: c:\documents and settings\cmartell\my
documents\my sas files\9.1\sesug07\myxml2.xml.
```

Figure 4 - XML with more nested levels generates error

We try reading the XML file in MS Access, hoping to find an acceptable intermediary. Two tables are created, no errors are generated, and all seems well until we realize that the relationships inherent in the XML structure are lost. Viewing the XML, it is easy to see the school where each class is located. This class/school relationship is lost in the

MS Access tables created, as seen in **Figure 5**. The *class* table needs a linking variable showing the school name. SAS XML Mapper enables you to retain such relationships.



school : Table

name	address	city	state
Grayson Eleme	123 Safe Neight	Anywhere	NC
Bayside Elemer	246 Dockside L	Lakefront	FL

Record: 1 of 2

class : Table

teacher	grade	room
Mr Jones	First	101
Ms Smith	Second	105
Mr Grady	First	103
Mr Flounder	Second	21
Ms Turtle	First	17
Ms Ocean	Third	11

Record: 1 of 6

Figure 5 - XML imported into MS Access tables

SAS XML MAPPER TO THE RESCUE

The SAS XML Mapper can be found on the Client-Side Components Volume 1 CD in version 9.1.3. (This tool was formerly called XML Atlas; the XML libname engine first appeared in v8.1.) It installs as a separate application, and is invoked outside of SAS. Online help files are included with the application and accessible from the application's Menu bar. When opened, the application appears as shown in **Figure 6**.

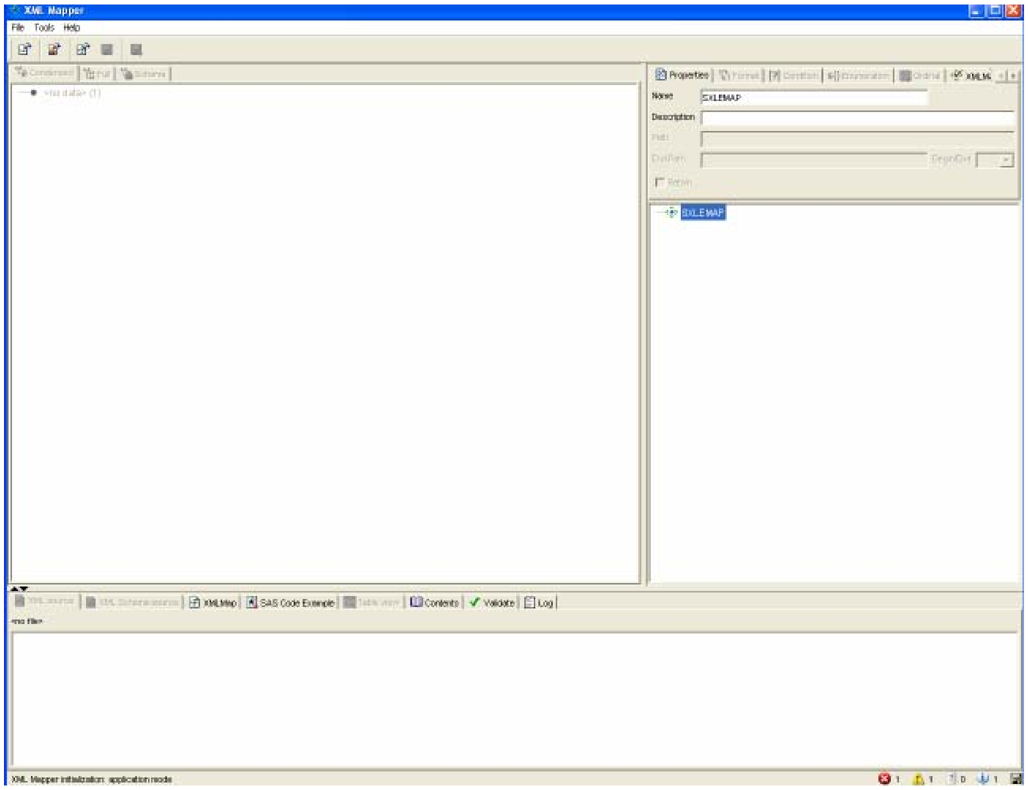


Figure 6 - XML Mapper opening screen

When an XML file is opened, the XML Mapper displays the XML nesting structures in the large pane on the left in an expandable tree view. Through the use of the tabs in the left pane, both a condensed view displaying only the structure and a full view revealing actual data values are available. Although they cannot actually be viewed simultaneously in the XML Mapper, the two views are displayed side-by-side here in **Figure 7**.

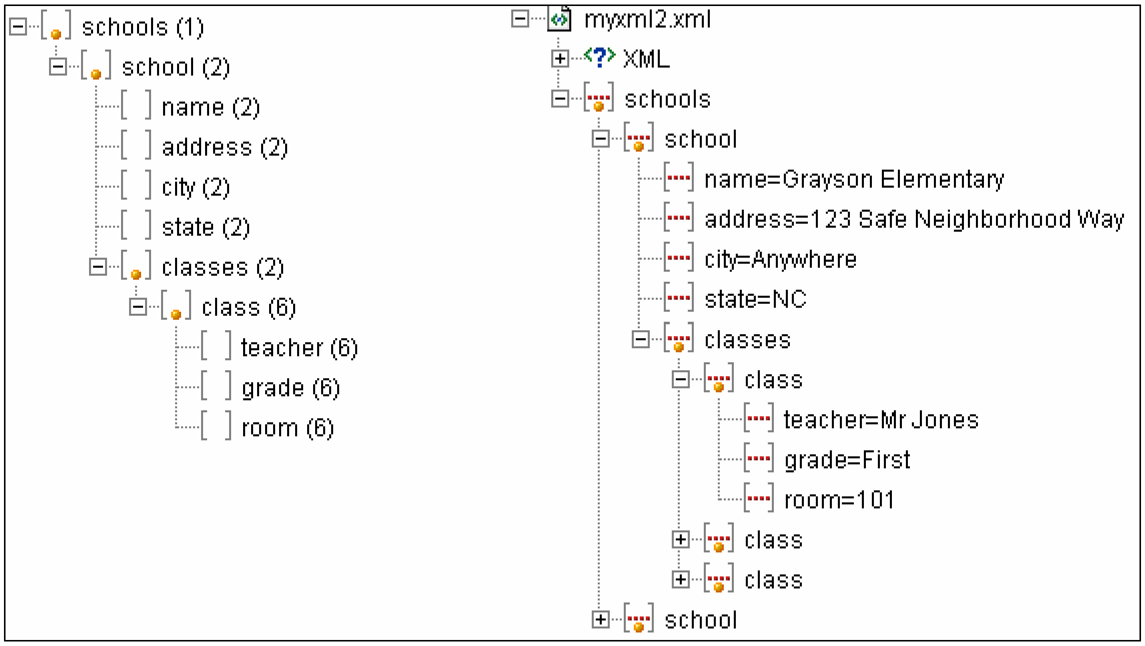
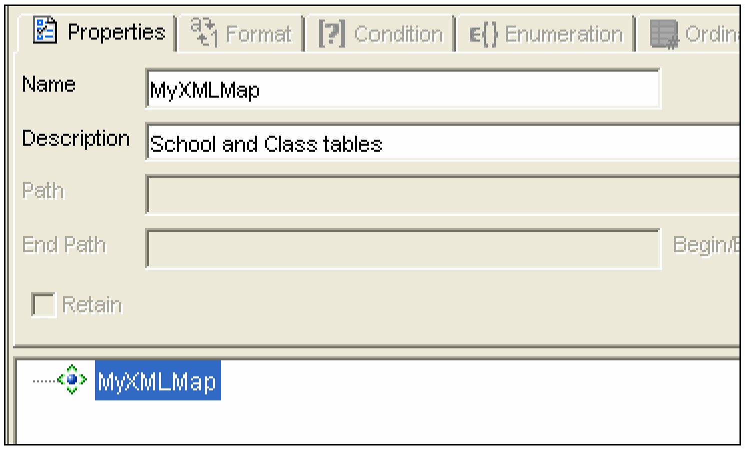


Figure 7 - Both condensed and full views of XML data

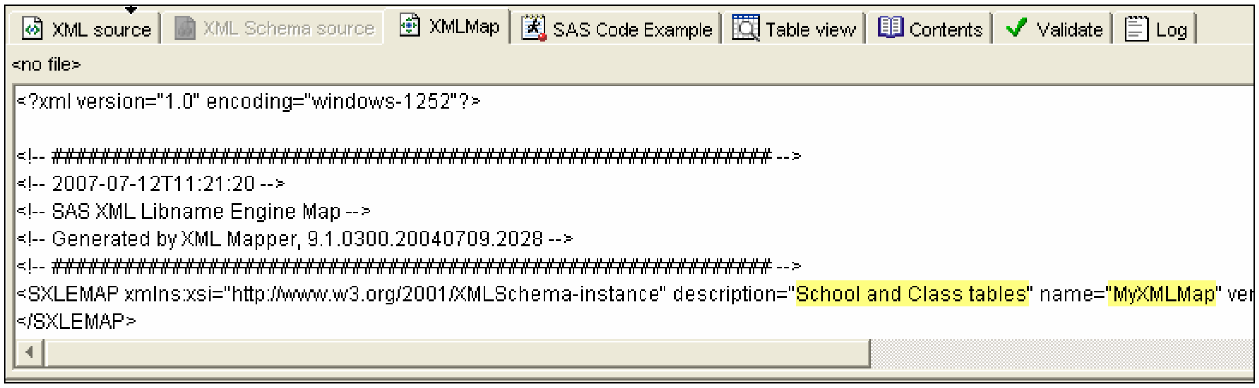
The right-hand pane is the work area for building an XMLMap. This tabbed pane is where you are given the means to dictate table structure (the formats and other tools in this pane are not addressed in this paper.) The bottom tabbed information pane can show the raw XML data, the XMLMap, sample SAS code for using the XMLMap, SAS views of the tables that will be built from the XMLMap, variable information from `PROC CONTENTS` for the tables, or validation information for the XMLMap. The information updates as you build the XMLMap.

We begin by naming and describing our XMLMap, as seen in **Figure 8**. Selecting the XMLMap tab in the information pane, we see in **Figure 9** that the name and description are now incorporated into the XMLMap syntax.



Name: MyXMLMap
Description: School and Class tables

Figure 8 - Naming the map



```
<?xml version="1.0" encoding="windows-1252"?>

<!-- ############################################################ -->
<!-- 2007-07-12T11:21:20 -->
<!-- SAS XML Libname Engine Map -->
<!-- Generated by XML Mapper, 9.1.0300.20040709.2028 -->
<!-- ############################################################ -->
<SXLEMAP xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" description="School and Class tables" name="MyXMLMap" ver
</SXLEMAP>
```

Figure 9 - XMLMap with name and description

We want both a *school* and a *class* table. We drag ‘school’ from the left pane and drop it onto ‘MyXMLMap’ in the right-hand pane. Alternatively, right-mouse-clicking items reveal popup menus – you can ‘get’ from the left pane and ‘put’ to the right pane, as shown in **Figure 10**. The dropped item appears indented beneath its target.

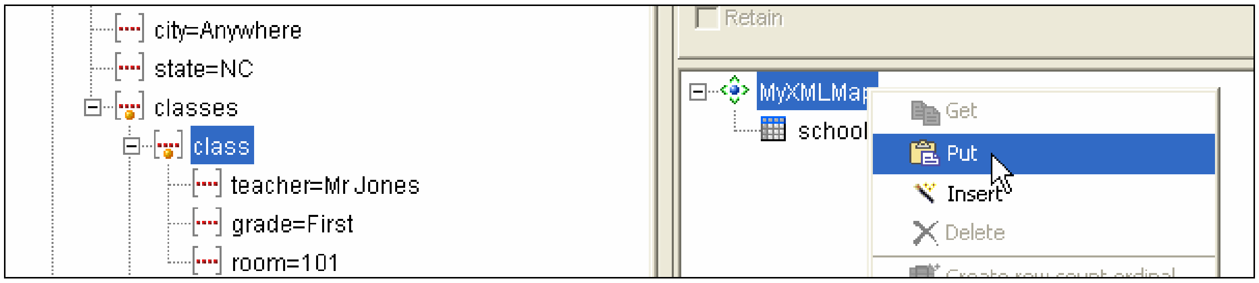
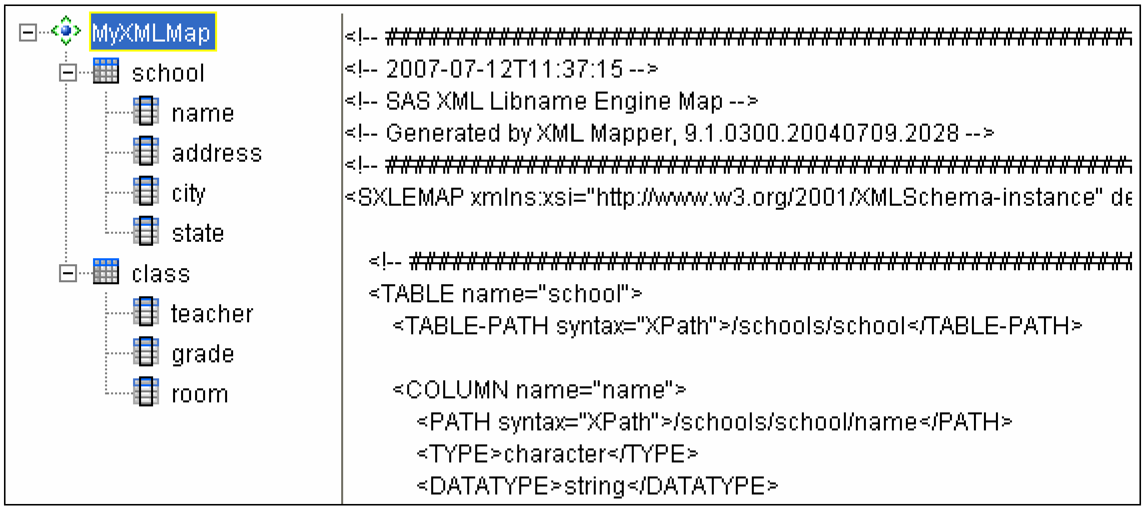


Figure 10 - Moving table names to the XMLMap

Because the XML Mapper is building a model for SAS tables, there can only be two levels under the map name: tables and variables. Items dropped on the map name become tables, and items dropped onto table names become variables. We drag all the variables from class and school in the XML structure into the map, placing them appropriately. We see the map structure and part of the XMLMap syntax from the information pane side-by-side in **Figure 11**. The information pane shows columns nested within tables and column attributes nested within columns.



```
<!-- ############################################################
<!-- 2007-07-12T11:37:15 -->
<!-- SAS XML Libname Engine Map -->
<!-- Generated by XML Mapper, 9.1.0300.20040709.2028 -->
<!-- ############################################################
<SXLEMAP xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" de

  <!-- ##########################################################
  <TABLE name="school">
    <TABLE-PATH syntax="XPath">/schools/school</TABLE-PATH>

    <COLUMN name="name">
      <PATH syntax="XPath">/schools/school/name</PATH>
      <TYPE>character</TYPE>
      <DATATYPE>string</DATATYPE>
```

Figure 11 - Table structure and XMLMap syntax

At this point, we have the same table structure as we had from the MS Access import. We can view the tables in the information pane at the bottom. The *class* table is shown in **Figure 12**. We are still lacking the school name.

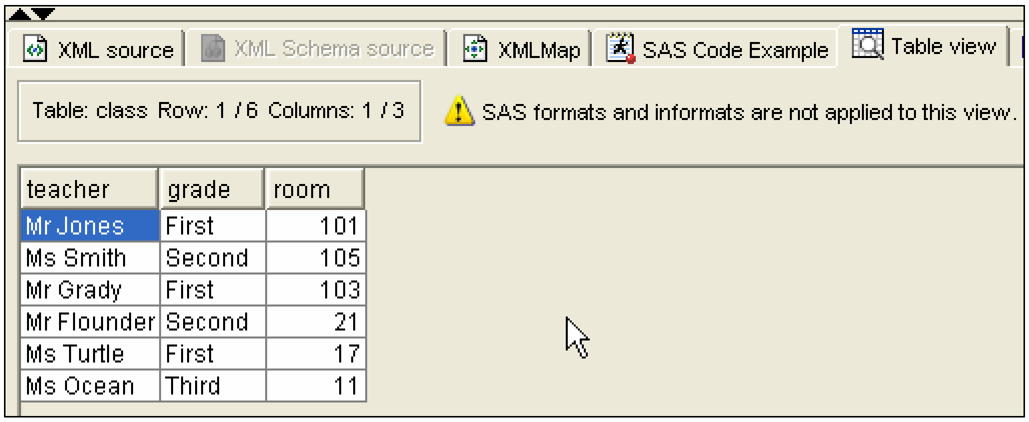


Table: class Row: 1 / 6 Columns: 1 / 3

SAS formats and informats are not applied to this view.

teacher	grade	room
Mr Jones	First	101
Ms Smith	Second	105
Mr Grady	First	103
Mr Flounder	Second	21
Ms Turtle	First	17
Ms Ocean	Third	11

Figure 12 - Class table in information pane

To retain the relationship information between classes and schools, we add school name to the table *class* by dragging the name field from the school variable list in the left pane and dropping it on the table *class* in the right pane. The new table structure and the modified table *class* appear together in **Figure 13**. The name of the parent school now appears in the table *class*. The SAS XMLMap has associated the appropriate school name with each class. Satisfied with our new table structure, we click the Validate tab in the information window in **Figure 14**.

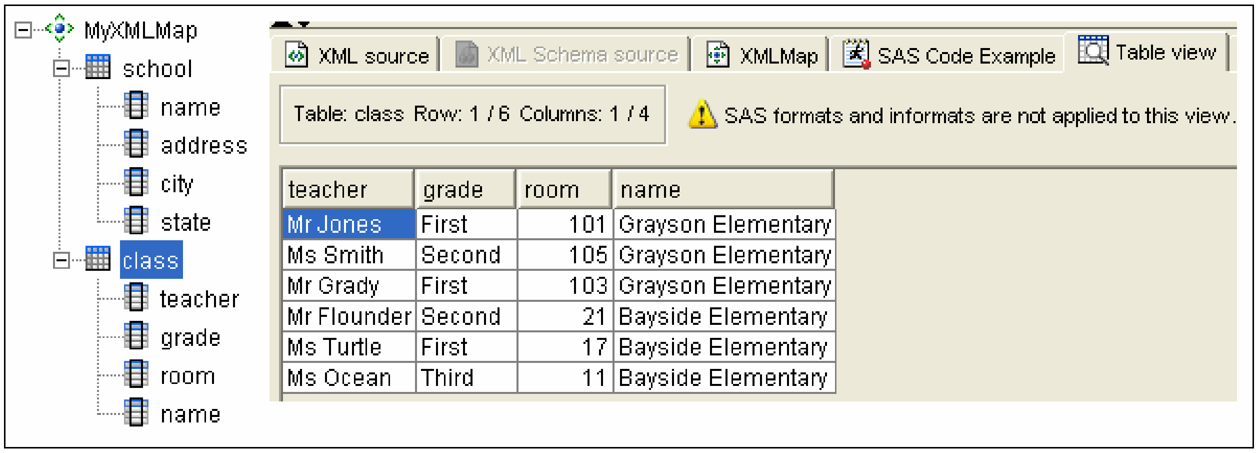
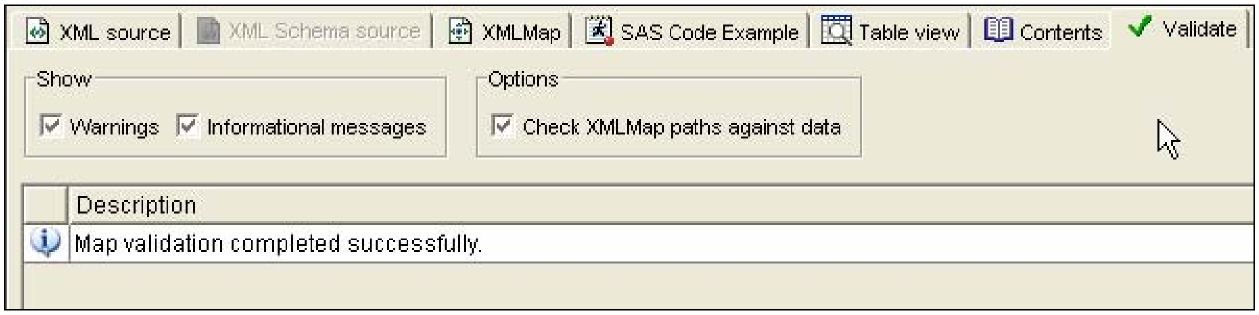


Table: class Row: 1 / 6 Columns: 1 / 4

SAS formats and informats are not applied to this view.

teacher	grade	room	name
Mr Jones	First	101	Grayson Elementary
Ms Smith	Second	105	Grayson Elementary
Mr Grady	First	103	Grayson Elementary
Mr Flounder	Second	21	Bayside Elementary
Ms Turtle	First	17	Bayside Elementary
Ms Ocean	Third	11	Bayside Elementary

Figure 13 - Class table with school name incorporated



Show: ☑ Warnings ☑ Informational messages

Options: ☑ Check XMLMap paths against data

	Description
ℹ	Map validation completed successfully.

Figure 14 - The XMLMap has no errors

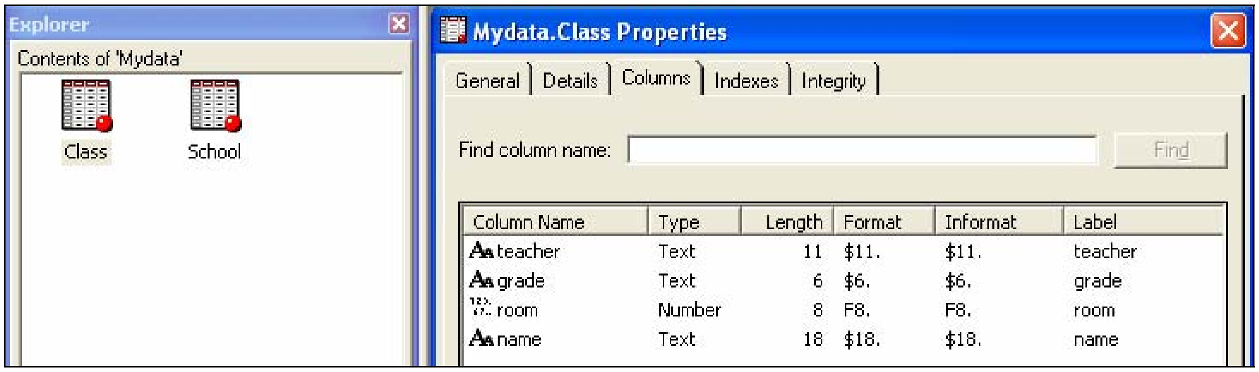
We save the XMLMap to myxml2.map, using the File menu in the SAS XML Mapper. This should enable us to read the file myxml2.xml. We modify the LIBNAME statement, that had previously generated an error message, adding an XMLMAP= parameter.

```
LIBNAME MYDATA XML 'C:\...\MYXML2.XML' XMLMAP='C:\...\MYXML2.MAP';
```

This time there is no error. The log shows:

```
NOTE: Libref MYDATA was successfully assigned as follows:
      Engine:        XML
      Physical Name: C:\...\MYXML2.XML
```

In **Figure 15** we see that the explorer window shows both tables in the library, and the columns for the table *class* now include the school name.

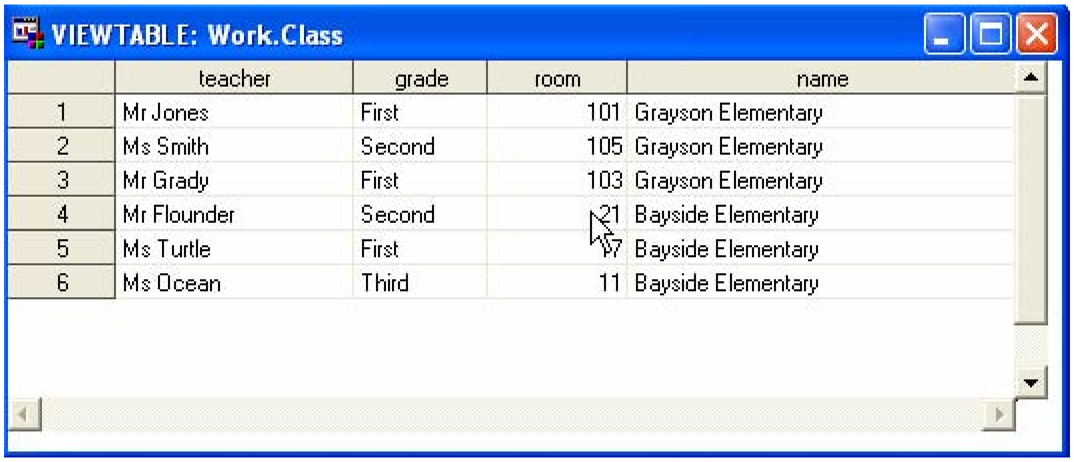


Column Name	Type	Length	Format	Informat	Label
teacher	Text	11	$11.	$11.	teacher
grade	Text	6	$6.	$6.	grade
room	Number	8	F8.	F8.	room
name	Text	18	$18.	$18.	name

Figure 15 - XML library view and class table properties

We can now run procedures that directly read this data. Since we did not specify ACCESS=READONLY in the LIBNAME statement, to interactively view the data we must copy it from the XML library. The resulting table *class* is shown in **Figure 16**.

```
DATA CLASS; SET MYDATA.CLASS; RUN;
```



	teacher	grade	room	name
1	Mr Jones	First	101	Grayson Elementary
2	Ms Smith	Second	105	Grayson Elementary
3	Mr Grady	First	103	Grayson Elementary
4	Mr Flounder	Second	21	Bayside Elementary
5	Ms Turtle	First	17	Bayside Elementary
6	Ms Ocean	Third	11	Bayside Elementary

Figure 16 – The class table copied into a work table

CONCLUSION

We have seen that SAS requires generic XML nesting to be limited to levels reflecting only tables and variables. Deviations from such a model in an XML file necessitate the use of a SAS XMLMap to extract data. The SAS XML Mapper provides a drag-and-drop interface for building the XMLMaps that define the table structures. In our example, we followed a relational model, creating two SAS tables, each containing a linking variable. We could as easily have built a single table containing both the class-level variables and the school-level variables. The SAS XML Mapper provides great flexibility while enabling you to painlessly build an XMLMap.

ACKNOWLEDGMENTS

Thomas Cox, lead developer for XML Mapper, SAS Institute Inc.

RECOMMENDED READING

The SAS 9.1.3 OnlineDoc contains information about the SAS XML Mapper. In the Contents, it is found as follows: SAS OnlineDoc/Where to Go for Additional Documentation/Pointers to Other SAS Products/SAS XML Mapper. It can also be located through various links found under the Base SAS XML LIBNAME Engine: User's Guide. An XML Engine topic can be found at http://support.sas.com/rnd/base. Under this topic, the latest version of SAS XML Mapper is referenced with version 9.1.3; a 9.1 version appears under SAS 9.1 XMLMap Tools; the precursor, XML Atlas, is referenced under SAS 9.0. A hotfix for SAS 8.2 is available to upgrade SAS 8.2 XML LIBNAME engine to SAS 9.1 production quality.

CONTACT INFORMATION

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

Carol A. Martell
UNC Highway Safety Research Center
730 MLK Jr. Blvd
Chapel Hill, NC 27514
Work Phone: 919-962-8713
Fax: 919-962-8710
E-mail: carol_martell@unc.edu