The Easy Way to Include XML Data into your SAS reports South Central SAS® Users Group Educational Forum 2011

Mary Grace Crissey, Pearson, San Antonio, TX, USA

ABSTRACT

Originally designed to meet the challenges of large-scale electronic publishing, XML is playing an increasingly important role in the exchange of a wide variety of data on the Web and elsewhere. Many heterogeneous information systems have chosen XML as their preferred method of data exchange - especially in pharmaceutical and medical environments, with usage growing in the financial and educational domains.   This paper illustrates a case study when I was given student test records and school demographic information as an XML file and asked to run standard SAS reports based on those inputs. With the stand alone utility tool , SAS XML Mapper 9.21, I was able to unlock the mystery of the XML "foreign" data and its hidden metadata structure . Interactively running the XML Mapper tool, I am able to visually ensure the hierarchical tree structure conforms to the given schema and thus come up with the syntax necessary to build a map to feed into the SAS Libname engine.   Snippets from my SAS code for educational testing assessment reporting project for the State of Wyoming are displayed in my PowerPoint as the SAS XML Mapper enables drag and drop exploration of the tags and Xpaths with XML. Familiarizing myself with data was essential, because I must report on ***all*** data be they flat files, MS excel spreadsheets, mainframe files, SAS data, and including those "dreaded" eXtensible Markup Language files.

INTRODUCTION

As a 20 year SAS veteran, I am comfortable with manipulating SAS datasets to generate insightful reports and perform advanced analytics. But when handed huge verbose XML files, I became confused because they appeared to be "gobble-de-gook" . For the past decade, quality assurance testers in my department have always received student test data and school demographic information either as mainframe "flat" files or as spreadsheets. It wasn't until I put on my SAS corrective lenses - namely SAS XML Mapper - that I was able to "see" clear enough to sort out variables and observations from the given XML File and load them into the XML Libname Engine. The strange unfamiliar looking data was again "friendly" when translated into a SAS dataset. Best of all, I can assure my employer, Pearson, that their contractual mandates for reporting are being met with the exact same Proc Tabulate and Proc Report statements as before so the displayed summaries and statistics are laid out in the predefined fashion according to the prescribed business rules.

XML AS SEEN FROM MY SAS PERSPECTIVE

XML roots go back to 1986, as it is a simple dialect of Standard Generalized Markup Language (SGML). Originally designed for publishers, technical writers and library automation personnel for catalog building and publishing of technical documents, it was by necessity quite complex and difficult to learn. Markup Languages were rarely adopted until a version referred to as Hypertext Markup Language (HTML) proliferated across the Internet . As large-scale electronic publishing demands grew , the number of predefined HTML markup tags became inadequate - so to fix this matter, XML was developed.

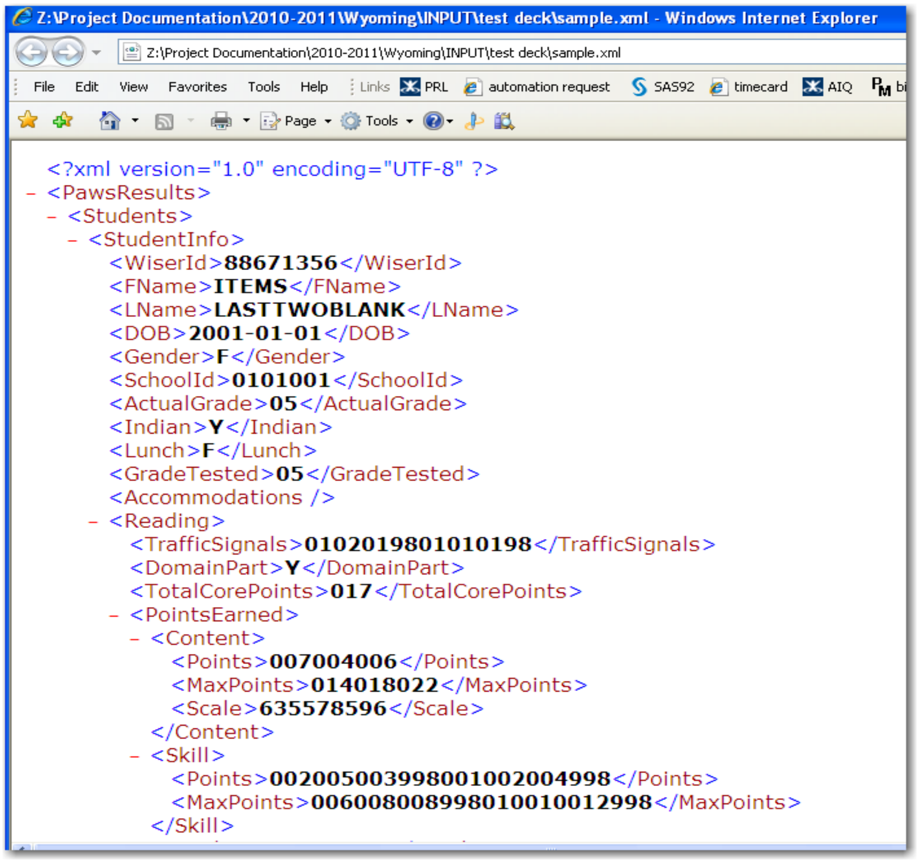
Since the release of the first version of Extensible Markup Language (XML) by the World Wide Web Consortium (W3C) in 1998, XML continues to gain popularity across industries as an integrating platform for applications and data sources. Not only content management and publishers, but also pharmaceutical industries and government agencies are leveraging XML technology. Today data storage vendors, warehousing and data mining providers are adopting XML standards into their data exchange systems to promote interoperability.

With the growing numbers of industries and implementations of XML across the world wide web, SAS customers, Data Experts, and forward looking IT providers must comprehend and properly account for complete information embedded within XML, namely the associated XML Schema (structure) as the Document Type Definition (DTD) as well as the values of actual data elements themselves.

SAS version 8 provided users a way to read XML documents with the Libname engine . This addressed "Simple and generic" cases of XML data. For the cases where XML did not automatically get read into SAS datasets via the XML engine , significant data step manipulations and SAS coding were required to define the necessary relationships and adequately pass XML information into SAS. Some may be comfortable enough with XML to use it natively and even produce style sheets for their XML formatted output. (XSL). In the references section of this paper, you will find names of SAS "pioneers" in the XML arena along with their related papers on their successful endeavors with XML.

With SAS 9 and SUGI20, workshops cropped up highlighting a new "XML mapper utility tool". This was the answer to my challenge, and with SAS XML Mapper, I was finally able to get SAS to successfully interpret my file. Thus the reason for this paper - to promote the good news that there is a stand alone Java application, that will automatically walk you thru the steps so you can generate your own custom XML MAPS for SAS XML engine. The examples given in my talk were done on a Windows system running Base SAS 9.2.

Since XML is text, this is what I saw when I "opened" the file given to me to analyze. I have worked with WebPages before and seen the HTML source code, so I could figure out some of what I was looking at - but I had no idea how to massage the data to get it where I could calculate the average scores for the students who took a particular test and then display it by grade and school district.


Z:\Project Documentation\2010-2011\Wyoming\INPUT\test deck\sample.xml - Windows Internet Explorer
File Edit View Favorites Tools Help
Links PRL automation request SAS92 timecard AIQ
Page Tools

```
<?xml version="1.0" encoding="UTF-8" ?>
- <PawsResults>
  - <Students>
    - <StudentInfo>
        <WiserId>88671356</WiserId>
        <FName>ITEMS</FName>
        <LName>LASTTWOBLANK</LName>
        <DOB>2001-01-01</DOB>
        <Gender>F</Gender>
        <SchoolId>0101001</SchoolId>
        <ActualGrade>05</ActualGrade>
        <Indian>Y</Indian>
        <Lunch>F</Lunch>
        <GradeTested>05</GradeTested>
        <Accommodations />
      - <Reading>
          <TrafficSignals>0102019801010198</TrafficSignals>
          <DomainPart>Y</DomainPart>
          <TotalCorePoints>017</TotalCorePoints>
        - <PointsEarned>
          - <Content>
              <Points>007004006</Points>
              <MaxPoints>014018022</MaxPoints>
              <Scale>635578596</Scale>
            </Content>
          - <Skill>
              <Points>002005003998001002004998</Points>
              <MaxPoints>006008008998010010012998</MaxPoints>
            </Skill>
```

Figure 1 - 107 students data in *.XML (1740 kb), the same data in *.xls (96K)

LOOKS LIKE HTML , SMELLS LIKE A SIMPLE TEXT FILE, BUT SURE TAKES ALOT OF SPACE

1. XML is text. All XML is text, accessible in a simple text editor. In SAS datasets, by contrast, fields can be text or numeric and, despite the rich set of text-processing functions in SAS, numeric data is easier to process than text data. In addition, proprietary tools such as SAS Universal Viewer or the base SAS product are necessary to access data and attribute information in a SAS dataset.

2. XML consists of named content. The name of each item in an XML file is written completely twice in text - when that element's scope begins and written out again completely when that element's scope ends. The sequence of items in that scope is predictable to some extent, but not fixed like it is for a typical SAS dataset. In addition, field attributes, such as field name, are not explicit as they are in SAS but in XML are stored separately from the data itself

3. XML is hierarchical, unlike SAS formatted input data.

   a. In the typical SAS setting, data exist in fields(variables) on records(observations) in datasets. In a given dataset, every record has precisely the same fields with precisely the same attributes. One relates a data item to another data item by the fact that they share, or do not share, key variables and key variable values.

   b. In XML, a data item relates to other data items by the ancestors, descendants, and siblings that they share. XML is hierarchical so a data item inherits identity from its ancestors, shares identity with its siblings, and passes on identity to its descendents. To identify a data item, one has to literally traverse all of its ancestors. This traversal creates a path through the XML file. By contrast, in the typical SAS dataset, one identifies a data item by looking at key fields on the same record.

SAS ACCEPTS DATA IN ALL SORTS OF SHAPES & SIZES

Not only is SAS recognized as the leader in statistical software, SAS boasts premier data integration systems and award winning business intelligence offerings. To me, the real power lies in how SAS ties it all together with a fully functional programming environment. The tremendous cross-platform capabilities of SAS is precisely why growing enterprises with heterogeneous information systems involving complex human-machine and machine-machine interactions are choosing SAS.

SAS is said to be 'data agnostic' meaning that it can accurately read (and write) over a dozen data storage formats. There are many ways to interact with SAS, and more than one "right" way to accomplish a task. When it comes to accessing raw data, different operating systems or legacy practices may make one method easier to run with - or particular SAS users may simply prefer a favorite technique for accessing raw data over another. External data may be found arranged as rows of digits and characters laid out in columns or free formatted. Raw data can be read in with SAS statements, SAS Functions, External File Interface(EFI) and the Import Wizard. In stream data can be read in with a DATA step, via the INPUT, DATALINES, and INFILE statements. One of my favorite reasons to go to Enterprise Guide is to take advantage of the easy to use import wizards, especially when moving data from an Microsoft excel spreadsheet to SAS.

The answer to dealing with XML data came with the XML LIBNAME Engine. Now SAS made it possible to translate XML markup to SAS proprietary format as shown in the excerpt from Documentation below

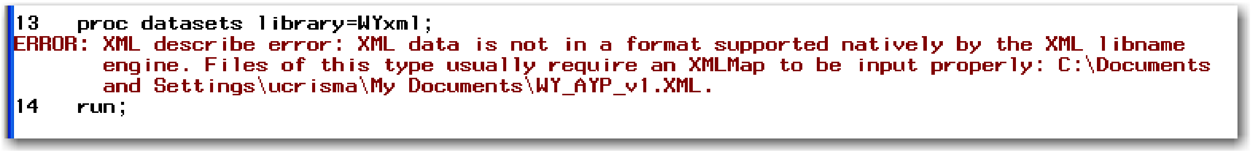
```
LIBNAME TRANS XML  'XML-document';
LIBNAME MYFILES 'SAS-library';
DATA MYFILES.CLASS;
SET TRANS.CLASS;
RUN;
```

The first LIBNAME statement assigns the libref TRANS to the physical location of the XML document (complete pathname, filename, and file extension) and specifies the XML engine. Not that the XML engine expects *GENERIC markup*. The second LIBNAME statement assigns the libref MYFILES to the physical location of the SAS library that will store the resulting SAS data set. The V9 engine is the default. The DATA step reads the XML document and writes its content in SAS proprietary format.

According to that syntax I took my data from Fig 1 and wrote my LIBNAME statement to apply the XML engine as follows:

```
LIBNAME WYXML XML 'C:\...\WY_AYP_V1.XML';
PROC DATASETS LIBRARY=WYXML; RUN;
```

But alas -- It did not run. My log displayed an ERROR message indicating that my particular XML file was neither "simple" nor "Generic" Markup Language .



```
13   proc datasets library=WYxml;
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\ucrisma\My Documents\WY_AYP_v1.XML.
14   run;
```

Figure 2 - ERROR message unsupported XML

Looking back at Figure 1, you will notice that the WY student data is nested at several levels. The first level contains information describing the student who took the test. The next level in describes the particular test the student took - for example the number of questions in the reading section followed my points in various skills measured on the reading test.

Before I try linking my XML file to the SAS LIBNAME engine, I consulted a detailed Schema that WY provided with the XML student file . This Schema declares the allowable values in the layout of the student file and can be into the XML mapper utility as input along with the XML data. Interactively running the java application I was able to stepping through and visually make sure variables and observations were ported correctly as an XML map. I named the output from the SAS XML Mapper as a file called “may31.map”   .

```
filename   WYp1 'C:\....\WY_AYP_V1.xml';
filename   SXLEMAP 'C:\....Wyoming\may31.map';
libname    WYp1 xml xmlmap=SXLEMAP access=READONLY;
```

This time I was successful . The first filename identified the XML file with the actual student data (WY_AYP_V1) and the other filename statement identified the XML Map for the LIBNAME engine to use when reading my XML student records. The may31.map file ensured that values imported as SAS observations from the LIBNAME engine would be accurate following the appropriate structure. Now I was able to recognize familiar variables and WY student records that looked like the files I had been getting as VSAM mainframe files. Today instead of reading them in with Filename reference and the INFILE statement in the data step, I associate the schema to the XML Mapper utility and create my SKLEMAP to feed into the the LIBNAME engine.

XML Mapper is designed to make it easy by automatically analyzing the structure of an XML file (or follow your given XML Schema) before it generates the necessary metadata. The process consolidates all instances of a specific Xpaths into a unique candidate field, counts the number of occurrences, captures maximum field length, keeps a representative sample of data instances, and heuristically suggests a data type (string, numeric, date, etc.) based on the sampling. The availability of this information makes identifying table, row, and column components much simpler. The drag-and-drop graphical interface reduces the time and effort to create an XMLMap and helps eliminate syntax errors, allowing the data analyst to focus on solving the problem, not the details of the implementation. A short summary of the process I followed is given below.

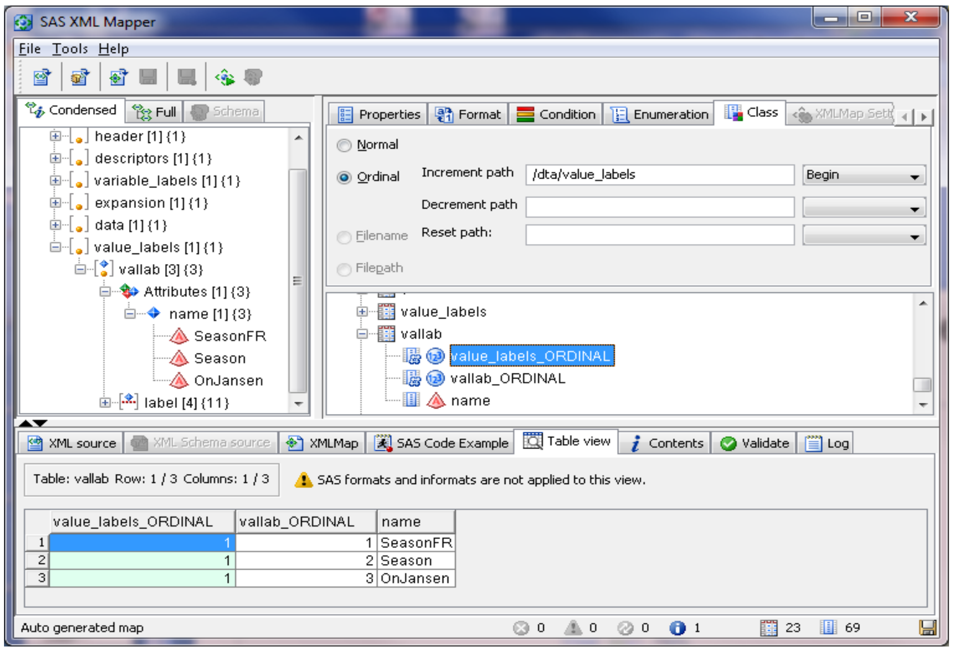
VIEWING COMPLICATED XML FILES

Step 1: Open your XML data files with XML MAPPER - The stand alone Java utility which permits you to SEE the xml data fields and corresponding values. (top left window)

Step 2: Link up your XML data to the corresponding schema by clicking the pull down menu under TOOLS and selecting AUTOMAP using xsd.

Step 3: When you “apply” your schema, the XML mapper utility will build you an XML map. This is what you need to pass to SAS so the XML Libname engine can accept input data as XML files and instruct SAS 9.2 (BASE) to transform the data into a SAS dataset.

Step 4 : Observe the Xpath names displayed in top right window. Drill down (click on the + sign to the left of the Field names) to see the properties and formats associated with each. Adjust as desired(Dates, char lengths, etc)



```
header [1] {1}
descriptors [1] {1}
variable_labels [1] {1}
expansion [1] {1}
data [1] {1}
value_labels [1] {1}
  vallab [3] {3}
    Attributes [1] {3}
      name [1] {3}
        SeasonFR
        Season
        OnJansen
    label [4] {11}
```

Normal
Ordinal — Increment path: /dta/value_labels — Begin
Decrement path:
Filename — Reset path:
Filepath

```
value_labels
vallab
  value_labels_ORDINAL
  vallab_ORDINAL
  name
```

Table: vallab Row: 1 / 3 Columns: 1 / 3

SAS formats and informats are not applied to this view.

	value_labels_ORDINAL	vallab_ORDINAL	name
1	1	1	SeasonFR
2	1	2	Season
3	1	3	OnJansen

Auto generated map 0 0 0 1 23 69

Figure 3 - Graphical User Interface for Creating XML maps for SAS XML Libname Engine

HOW TO FIND OUT IF YOUR SITE HAS A PRODUCT LICENSE FOR XML MAPPER?

Before you can apply any new SAS tip or Trick shown at this educational forum (or any user group meeting) you best first check and see if additional site licenses for extra software products are required. As always the best way to figure out what SAS products are available on your IT site is to run the "**Proc Setinit**" command. Doing so , your log file will list the names of all the products (with version numbers) --- that are on your site license with the dates of your license renewal period. While you will **not** see SAS XML Mapper listed, do not be alarmed. Certainly don't worry about how to make a budget request at the next management budget proposal review. All SAS customers (who have Base SAS) have full access to run with XML Mapper .

How Do I Get SAS XML Mapper?

This is the amazing part - SAS is "giving away" this tool. If your site administrator did not install the java stand alone application on your machine for you - and you do not have the installation media for Windows and UNIX platforms, You can download it yourself from the SAS Support site. I am no expert myself at installing software, but the instructions in the readme files accompanying the utility are easy to follow. While there is a Book version of the SAS documentation available from the SAS bookstore for SAS XML Libname which includes a chapter on the XML mapper utility - I found the usage examples given in the Online Help much more instructional.

Navigating the SAS support pages

Knowing that it is possible to unlock the mystery of the XML "foreign" data and its hidden metadata structure is only beneficial to you if you can get the stand alone tool installed on your computer. The trick is finding where the Java application is hiding on the SAS support WebPages. Starting at their Hot fix and Download page http://support.sas.com/techsup/dwnload, click on the top bullet in the download section that says *SAS software - SAS Product and Solution updates.* From here you will see a laundry list of SAS offerings, and since this utility is associated with the BASE SAS product ,you will click on BASE SAS Software (top right column entry). Take caution on the next page because there are many old experimental versions of software available to download - so make sure you start at the top and get the most recent information next to the icon that says "BASE SAS ROCKS" .

CONCLUSION

This obstacle I faced when trying to “read” my XML data before passing to my canned SAS programs as required to generate educational assessment reports was daunting. Happily, as Carol Martell said in her 2008 Global Forum paper ““SAS® XML Mapper came To the Rescue”. Drag and Dropping with the interactive Java application (and several exchanges with SAS Technical Support), I can now successfully produce reports from the large verbose XML data that WYOMING Department of Education sends us to analyze.

Last month , SAS released more enhancements to the XML Libname engine and SAS XML Mapper utility with version 9.3 . The newest user guide for XML mapper utility says that SAS 9.2 users can run the latest utility even before they install SAS 9.3 I have not explored the upgrades yet, but look forward to learning better ways to quickly investigate and solve my data analysis projects.

Many XML-based initiatives are under way to formulate models and define standards to unify and speed up the practices. As companies recognize business benefits derived from XML solutions,SAS users should become familiar with the ways to import XML into SAS, export XML from SAS, as well as learn how to transform one XML instance into another.

In 2011, my task for Pearson was to validate the XML files and produce deliverables as pdf reports. The actual validating involves combining XML files with Merges, Sorts and Extracts and “Proc Compares” of files from several agencies in various locations. The preferred style for passing along the “business intelligence” (BI) of student score reports and educational system assessments may be pdf files today, but in the future, state education agencies might request their BI reports be “sent” via XML. Armed with SAS export wizards, ODS, DI systems and continual support we can confidently respond by adapting to future technology changes ahead.

REFERENCES

Bos, Bert (1999) “XML in 10 Points” available at http://www.w3.org/XML/1999/XML-in-10-points

Castro, Elizabeth. (2001) “XML for the World Wide Web”. Peachpit Press: Berkeley, CA

Foley, Richard , Friebel Anthony (2008) “Power Up SAS® with XML” , Proceedings of SAS Global Forum 2008. Cary, NC:SAS Institute, Inc.

Friebel, Anthony. (2003) “XML? We do that!”. Paper 173-28. SUGI 28 Proceedings. http://www2.sas.com/proceedings/sugi28/173-28.pdf

Hoyle, Larry (2010) “Using XML Mapper and Enterprise Guide to Read Data and Metadata from an XML File”, paper 30, SAS Global Forum 2010 Proceedings, Cary,NC: SAS Institute, Inc.

Hoyle, Larry : Wackerow,Joachim (2008) “Exporting SAS Datasets to DDI 3 XML Files - Data, Metadata, and More Metadata” available at http://www2.sas.com/proceedings/forum2008/137-2008.pdf

Hoyle, Larry (2006) “Reading Microsoft Word XML files with SAS” available at http://www2.sas.com/proceedings/sugi31/019-31.pdf

Hoyle, Larry (2005) “Using XML Mapper and XMLMAP to Read Data Documented by Data Documentation Initiative (DDI) Files SUGI30 available at http://www2.sas.com/proceedings/sugi30/099-30.pdf

Jansen, Lex (2008) “Using SAS® XML Mapper and ODS PDF to create a PDF representation of the define.xml (that can be printed)” NESUG2008 proceedings http://www.nesug.org/Proceedings/nesug08/ph/ph06.pdf

Jansen, Lex (2010) “Accessing the metadata from the define.xml using XSLT transformations “ available at http://www.lexjansen.com/pharmasug/2010/cd/cd14.pdf

Martell, Carol.(2008) “SAS® XML Mapper To the Rescue”. SAS Global Forum Proceedings available http://www2.sas.com/proceedings/forum2008/099-2008.pdf

Milum, Jenine (2011) “Let’s Get Connected: How We Link to our Data” Paper 139- SAS Global Forum 2011

Nelson, Greg B. (2000) “XML and SAS®: An Advanced Tutorial” SUGI25 proceedings, Cary, NC:SAS Institute

Pratter, Frederick (2008) “XML for SAS® Programmers”, www2.sas.com/proceedings/forum2008/042-2008.pdf

Shoemaker, Jack N., Nelson, Greg B(2003) “ XML Primer for SAS® Programmers” SUGI 28 Proceedings http://www2.sas.com/proceedings/sugi28/193-28.pdf

RECOMMENDED READING

- SAS® 9.2 XML LIBNAME ENGINE: USER’S GUIDE http://support.sas.com/documentation/cdl/en/engxml/62845/PDF/default/engxml.pdf
- “Using SAS XML Mapper to Generate and Update an XMLMap” Chapter11 of SAS® 9.3 XML LIBNAME ENGINE: Users Guide http://support.sas.com/documentation/cdl/en/engxml/63177/PDF/default/engxml.pdf
- SAS® XML Mapper 9.3 INSTALLATION GUIDE http://support.sas.com/demosdownloads/000713/readme_110362.pdf
- SAS® XML Mapper 9.3 Software Download (Free for all BASE SAS users) http://support.sas.com/demosdownloads/sysdep_t1.jsp?packageID=000713&jmpflag=N

CONTACT INFORMATION

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

Mary Grace Crissey
Pearson
19500 Bulverde Rd
San Antonio, TX 78259
(210) 442-7428
marygrace38@gmail.com
www.linkedin.com/in/mgcrissey