Lab 6 – Semi-Structured Data

This is a one-week lab

The objective of this lab is to use the semi-structured representation of XML to represent the data in your project. We will use XML Schema to develop the structure of data, XML documents to represent the data instances and XSLT to provide reporting schemes on your data.

We will continue working on the project that you have developed so far in the previous labs, but this time we will change the data manipulation model from SQLite to a semi-structured representation.

In relational databases such as SQLite, the structure of the data that can be stored in each table is determined when the table is created. In other words, the CREATE TABLE command specifies how many columns and of what type each table has. So when inserting or retrieving data from the table, the DBMS would know which types of data are permissible. The same concept is applicable to XML documents as well. We can develop the so-called XML schemas that provide the structure for acceptable XML documents that would conform to these schemas.

The first step of this lab will be dedicated to converting your relational tables in SQLite into XML schemas.

I will continue to work on my movie database application. Up to now, I have two tables and would like to convert their structure into an XML schema structure.

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="movieData" type="movieType">

<xs:key name="dirKey">

<xs:selector xpath = "director"/>

<xs:field xpath = "directorName "/>

</xs:key>

<xs:keyref name="MovieFKey" refer="dirKey">

<xs:selector xpath = "movie"/>

<xs:field xpath = "dirName"/>

</xs:keyref>

</xs:element>

<xs:complexType name="movieType">

<xs:sequence>

<xs:element ref="movie" minOccurs="0" maxOccurs="unbounded"/>

<xs:element ref="director" minOccurs="0" maxOccurs="unbounded"/>

</xs:sequence>

</xs:complexType>

<xs:element name="director">

<xs:complexType>

<xs:sequence>

<xs:element name="directorName " type="xs:string"/>

<xs:element name="DOB" type="xs:string"/>

<xs:element name="Country" type="xs:string"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<xs:element name="movie">

<xs:complexType>

<xs:sequence>

<xs:element name="movieName " type="xs:string"/>

<xs:element name="Genre" type="xs:string"/>

<xs:element name="dirName" type="xs:string"/>

<xs:element name="budget" type="xs:decimal"/>

<xs:element name="productionYear" type="xs:decimal"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:schema>

The above XML schema specifies that I have two key entities, movie and entity. The movie entity has a director, the director’s name is always unique and that values for the director’s name would need to be constrained by those director names that are already defined by the director entity in the same XML document.

Once you have completed the conversion of your tables into XML schema format, you can copy it to the right hand side box of the following link and click ‘check XSD validity’:

<http://www.utilities-online.info/xsdvalidation/>

Now, I need to create an XML document with all the instances of data that I have in my movie and director tables. When creating the XML document, I need to make sure I remember to important consideration: 1) the integrity constraints that I have set are respected and 2) the XML document that I create conforms with the XML Schema.

<?xml version="1.0" encoding="UTF-8"?>

<movieData>

<movie>

<movieName>Inception</movieName>

<Genre>Mystery</Genre>

<dirName>Christopher Nolan</dirName>

<budget>160000000</budget>

<productionYear>2010</productionYear>

</movie>

<director>

<directorName>Christopher Nolan</directorName>

<DOB>July 30, 1970</DOB>

<Country>England</Country>

</director>

</movieData>

As an example, I have created one director and one movie instance. You can check whether your XML document is well formed by clicking on ‘check XML well formed’ button and also check whether your XML document is valid against your XML schema by clicking on ‘validate XML against XSD’.

You can copy and paste the above XML and XSD to the validator page and try it for yourself. Change Christopher Nolan in the dirName tag and see what happens. Why is that?

Now, that we have the XML schema and a valid XML instance document, we can use XSLT to produce various representations or customizations of the data.

Lets suppose that I further extend my XML instance document to the following:

<?xml version="1.0" encoding="UTF-8"?>

<movieData>

<movie>

<movieName>Inception</movieName>

<Genre>Mystery</Genre>

<dirName>Christopher Nolan</dirName>

<budget>160000000</budget>

<productionYear>2010</productionYear>

</movie>

<movie>

<movieName>Interstellar</movieName>

<Genre>Adventure</Genre>

<dirName>Christopher Nolan</dirName>

<budget>165000000</budget>

<productionYear>2014</productionYear>

</movie>

<movie>

<movieName>Avatar</movieName>

<Genre>Mystery</Genre>

<dirName>James Cameron</dirName>

<budget>237000000</budget>

<productionYear>2009</productionYear>

</movie>

<director>

<directorName>Christopher Nolan</directorName>

<DOB>July 30, 1970</DOB>

<Country>England</Country>

</director>

<director>

<directorName>James Cameron</directorName>

<DOB>August 16, 1954</DOB>

<Country>Canada</Country>

</director>

</movieData>

I would like to answer a few queries over this XML document. I can easily write appropriate XSLT to achieve this.

1. I would like to see the details of those movies whose director where born in Canada.

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">

<html>

<body>

<h2>Movies with Canadian Directors</h2>

<table border="1">

<tr bgcolor="#9acd32">

<th style="text-align:left">Movie</th>

<th style="text-align:left">Director</th>

</tr>

<xsl:for-each select="movieData/movie">

<tr>

<xsl:variable name="dirname" select="dirName" />

<xsl:variable name="directorData" select="//movieData/director[directorName=$dirname]" />

<xsl:if test="$directorData/Country='Canada'">

<td><xsl:value-of select="movieName"/></td>

<td><xsl:value-of select="dirName"/></td>

</xsl:if>

</tr>

</xsl:for-each>

</table>

</body>

</html>

</xsl:template>

</xsl:stylesheet>

1. I would like to see the details of all those directors that have directed an Adventure movie.

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">

<html>

<body>

<h2>Directors with Adventure Movies</h2>

<table border="1">

<tr bgcolor="#9acd32">

<th style="text-align:left">Director Name</th>

<th style="text-align:left">DOB</th>

<th style="text-align:left">Country</th>

</tr>

<xsl:for-each select="movieData/director">

<tr>

<xsl:variable name="dirname" select="directorName"/>

<xsl:variable name="movieInfo" select="//movieData/movie[Genre='Adventure' and dirName=$dirname] " />

<xsl:if test="$movieInfo/Genre">

<td><xsl:value-of select="directorName"/></td>

<td><xsl:value-of select="DOB"/></td>

<td><xsl:value-of select="Country"/></td>

</xsl:if>

</tr>

</xsl:for-each>

</table>

</body>

</html>

</xsl:template>

</xsl:stylesheet>

You can go to the following URL: <http://www.w3schools.com/xsl/tryxslt.asp?xmlfile=cdcatalog&xsltfile=cdcatalog>

and try the above XML document along with the XSLT queries to see the results.

For your lab, inspired by the above steps, you will need to perform the following tasks:

1. Convert the structure of the tables that you have created in the previous labs into an XML schema document. You should have created at least 10 tables by now, so it is expected that you will have 10 entities such as my movie and director entities in your XML schema.
2. Create a corresponding XML instance document with 5 instances for each of the entities in your XML schema. For instance, I have Christopher Nolan and James Cameron for the director entity and Inception, Interstellar, and Avatar. You need to have five instances for each of your entities.
3. Make sure your XML Schema is valid and your XML instance document is well formed. Also check that your XML instance document is valid with respect to your XML Schema.
4. Write 5 XSLT queries based on your entities and the data instances in your XML document. The queries can be similar to the SQL queries that you had in the previous labs or can be new queries that you think are interesting. Make sure that the results of your queries are not an empty set.

**Deliverables**

You should complete the steps described above. Then, you will prepare and submit the results in one single zip file (**YourName\_Lab6.zip**) containing the following item:

* You will need to submit the file for your XML scheme, as well as the XML instance document.
* Include a PDF document in which you will need to present the ER diagram that you had developed earlier in this course and explain briefly how the entities in your ER diagram have corresponding entities in your XML schema. Explain if you had to make any changes to your earlier design to make it more suitable for the XML schema design.
* Include one PDF file that consists of 5 queries in natural language and the corresponding XSLT queries. The TAs should be able to copy your XSLT into an XSLT tool along with your XML instance document and be able to replicate your results. For each query provide the following:
  + The query that you are designing. Similar to the description of the two queries I have included in this lab.
  + The XSLT query that you have designed for each of the queries
  + A screenshot of the results that you have obtained after running the XSLT over your XML instance document.