http://www.sqlservercentral.com/articles/XML/61290/
Printed 2007/10/31 04:22PM

XML Workshop IX - Mixed Types

By Jacob Sebastian, 2007/10/09

Introduction

In the last few installments of the XML Workshop, I had been trying to give a detailed explanation of some of the building blocks of *XSD*. There are a few more points that we still need to discuss before closing down the *XSD* section of the *XML Workshop*.

XML is getting widely accepted as the format for data exchange. Wide adoption of architectures like *SOA (Service Oriented Architecture)* contributed to the growth of applications and services that talk to each other and exchange data in XML format. When applications exchange data, there will be strict rules that define the structure as well as the quality of data. If the data does not adhere to the rules, it may not be useful at all.

XSD helps you to define those rules which will validate the structure and quality of the data that your application sends or receives. Complex validation rules can be broken down to simple *XSD* definitions and can be stored to an *XML SCHEMA COLLECTION*. You could then ask *SQL Server 2005* to validate the data based on the definitions in the *XML SCHEMA COLLECTION*.

In the previous few sessions, we have seen how to define rules which validate the structure of an *XML* document. We also learned how to define rules which validate the quality of data. Before we move out of *XSD*, I would like to present a few more *XSD* elements and attributes which define advanced characteristics of an *XML* document.

In the previous sessions, we have seen XML documents, composed of XML elements and attributes. We learned that if the element is a *Simple Type* it contains a value. If the element is a *ComplexType* then it will contain other elements and/or attributes. All the elements we have seen so far hold either a value or other child elements, but not both. In this session, we are going to have a look at *Mixed types*. *Mixed Type* is an element which will contain a value as well as other child elements.

Mixed Types

Mixed types are elements which contain value as well as other child elements. *Mixed types* usually contain long text data like the body of a letter, e-mail message or similar data where the data itself contains other specific pieces of information which needs to be extracted separately. The following example shows a case where a *Mixed Type* may be used.

SQLServerCentral sends an e-mail to the author when an article is scheduled for publication. Let us design a basic *XML* structure that holds the notification details. Here is a simple schema that holds the basic information. [schema1.xml]

```
1 <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
2   <xs:element name="ArticleNotification">
3     <xs:complexType>
4       <xs:sequence>
5         <xs:element name="Body" type="xs:string"/>
6         <xs:element name="Title" type="xs:string" />
7         <xs:element name="Url" type="xs:string" />
8         <xs:element name="ScheduledDate" type="xs:date"/>
9       </xs:sequence>
```

```
    </xs:complexType>
  </xs:element>
</xs:schema>
```

The above *SCHEMA* defines the following *XML* structure.

```
<ArticleNotification>
  <Body>The following article is scheduled for publication.</Body>
  <Title>XML Workshop VII - Validating values with SCHEMA</Title>
  <Url>http://www.sqlservercentral.com/3120.asp</Url>
  <ScheduledDate>2007-01-01Z</ScheduledDate>
</ArticleNotification>
```

Mixed Mode gives us the freedom to alter the *XML* structure to be more descriptively. For example, the following *XML* structure is preferable over the previous one in this specific case..

```
<ArticleNotification>
  Hi<Name>Jacob</Name>,
  Please note that your article
  <Title>XML Workshop VII - Validating values with SCHEMA</Title>,
  posted at <Url>http://www.sqlservercentral.com/3120.asp</Url> is
  scheduled for publication
  on <ScheduledDate>2007-01-01</ScheduledDate>.
</ArticleNotification>
```

If you would like to be a little more concise, you could go for a better structure as given below.

```
<ArticleNotification Url="http://www.sqlservercentral.com/3120.asp">
  Hi<Name>Jacob</Name>,
  Please note that your article
  <Title>XML Workshop VII - Validating values with SCHEMA</Title>
  is scheduled for publication
  on <ScheduledDate>2007-01-01</ScheduledDate>.
</ArticleNotification>
```

The XML structure I presented above is different from the structures that we have seen so far. In the above example, we have an element which has a value, an attribute and has other child elements. So, how do we define such elements in *XSD*? The answer is *"by using Mixed Types"*. Let us write the *XSD* for the above *XML* structure. [Download schema2.xml]

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="ArticleNotification">
    <xs:complexType mixed="true">
      <xs:sequence>
        <xs:element name="Name" type="xs:string" />
        <xs:element name="Title" type="xs:string"/>
        <xs:element name="ScheduledDate" type="xs:date"/>
      </xs:sequence>
      <xs:attribute name="Url" type ="xs:string" />
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Let us create the *SCHEMA* collection. [Download sql1.sql

```
CREATE XML SCHEMA COLLECTION ArticleNotificationSchema AS '
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="ArticleNotification">
    <xs:complexType mixed="true">
      <xs:sequence>
        <xs:element name="Name" type="xs:string" />
        <xs:element name="Title" type="xs:string"/>
        <xs:element name="ScheduledDate" type="xs:date"/>
      </xs:sequence>
      <xs:attribute name="Url" type ="xs:string" />
    </xs:complexType>
  </xs:element>
</xs:schema>
```

```
'
GO

/*
Let us test the schema by assigning a value with the
desired structure to an XML variable which is bound to the
above schema.
*/

DECLARE @art AS XML(ArticleNotificationSchema)
SET @art = '
<ArticleNotification Url="http://www.sqlservercentral.com/3120.asp">
  Hi<Name>Jacob</Name>,
  Please note that your article
  <Title>XML Workshop VII - Validating values with SCHEMA</Title>
  is scheduled for publication
  on <ScheduledDate>2007-01-01Z</ScheduledDate>.
</ArticleNotification>
'
```

Note that I have added "Z" at the end of the date value. If we don't do that, SQL Server will generate an error message. SQL Server 2005's implementation of *"xs:date"* data type takes Time Zone information along with a date value. SQL Server 2005 will not accept a date value without the Time Zone information. The following *XML* is equally valid. [Download sql2.sql]

```
DECLARE @art AS XML(ArticleNotificationSchema)
SET @art = '
<ArticleNotification Url="http://www.sqlservercentral.com/3120.asp">
  Hi<Name>Jacob</Name>,
  Please note that your article
  <Title>XML Workshop VII - Validating values with SCHEMA</Title>
  is scheduled for publication
  on <ScheduledDate>2007-01-01+05:30</ScheduledDate>.
</ArticleNotification>
'
```

Note the value *"+05:30"* which indicates Indian Standard Time, which is *GMT+05:30*.

Let us try to read information from the *XML* variable that we just created. Reading some of the information from the *XML* variable might seem little tricky. [Download sql3.sql]

```
DECLARE @art AS XML(ArticleNotificationSchema)
SET @art = '
<ArticleNotification Url="http://www.sqlservercentral.com/3120.asp">
  Hi<Name>Jacob</Name>,
  Please note that your article
  <Title>XML Workshop VII - Validating values with SCHEMA</Title>
  is scheduled for publication
  on <ScheduledDate>2007-01-01Z</ScheduledDate>.
</ArticleNotification>
'

/*
Let us make sure that the value is stored correctly.
Try to retrieve the "Url" attribute from the XML.
*/
SELECT
    x.a.value('(@Url)[1]', 'varchar(50)') AS Url
FROM @art.nodes('/ArticleNotification') x(a)

/*
OUTPUT:

Url
----------------------------------------
http://www.sqlservercentral.com/3120.asp

(1 row(s) affected)
*/
```

```
29
30 /*
31 Reading "name" and "title" is pretty simple.
32 */
33
34 SELECT
35     x.a.value('(Name)[1]', 'varchar(10)') AS [Name],
36     x.a.value('(Title)[1]','varchar(50)') AS Title,
37     x.a.value('(ScheduledDate)[1]','datetime') AS ScheduledDate
38 FROM @art.nodes('/ArticleNotification') x(a)
39
40 /*
41 OUTPUT:
42
43 Name   Title                                              ScheduledDate
44 ------ -------------------------------------------------- -------------------
45 Jacob XML Workshop VII - Validating values with SCHEMA 2007-01-01 00:00:00
46
47 (1 row(s) affected)
48
49 */
```

Good so far. Now how do we read the content of the *ArticleNotification* element? How do we retrieve the text from a *mixed element*? Here is the *TSQL* query which retrieves the whole text of the mixed element. [Download sql4.sql]

```
1  DECLARE @art AS XML(ArticleNotificationSchema)
2  SET @art = '
3  <ArticleNotification Url="http://www.sqlservercentral.com/3120.asp">
4    Hi <Name>Jacob</Name>,
5    Please note that your article
6    <Title>XML Workshop VII - Validating values with SCHEMA</Title>
7    is scheduled for publication
8    on <ScheduledDate>2007-01-01Z</ScheduledDate>.
9  </ArticleNotification>
10 '
11
12 /*
13 Read the entire text from the "ArticleNotification" element.
14 */
15
16 SELECT
17     x.a.value('(data(/))[1]', 'varchar(max)') AS Summary
18 FROM @art.nodes('/ArticleNotification') x(a)
19
20 /*
21 OUTPUT:
22
23 Summary
24 -------------------------------------------------
25
26   Hi Jacob,
27   Please note that your article
28   XML Workshop VII - Validating values with SCHEMA
29   is scheduled for publication
30   on 2007-01-01Z.
31
32
33 (1 row(s) affected)
34
35 */
```

We have not looked into *XQUERY* yet. The syntax and keywords used in the above queries may be little confusing to some of you. We will soon do a good review of *XQUERY* in the next few sessions.

Conclusions

In the last few sessions, we had been trying to understand *XSD* implementation of *SQL Server 2005* better.

There are a few more *XSD* points to discuss and then we will move to *XQUERY*.
