XML and JSON making simple changes

As part of the XML and JSON month of blog posts, this post is going to cover how to make changes to XML and JSON in TSQL.We will be using the same sample data from the original post with a data structure holding employee data.

```
CREATE TABLE XmlAndJson (  id INTEGER IDENTITY NOT NULL,  XmlData XML,
  JsonData NVARCHAR(MAX) ); GO ALTER TABLE XmlAndJson   ADD CONSTRAINT [J
sonData record should be formatted as JSON]   CHECK (ISJSON(JsonData)=1);
  GO SET NOCOUNT ON; DECLARE @XmlData as XML; set @XmlData = '        Ste
ve  Stedman           Derrick  Bovenkamp           Bill  Simpkins    ';
DECLARE @JsonData as VARCHAR(MAX); set @JsonData = ' {"employees":[   { "
firstName":"Steve", "lastName":"Stedman" },   { "firstName":"Derrick", "l
astName":"Bovenkamp" },   { "firstName":"Bill", "lastName":"Simpkins" } ]
}';  INSERT INTO XmlAndJson(XmlData, JsonData) VALUES (@XmlData, @JsonDat
a );
```

Since we are just past the 4th of July (Independence Day in the US), we will attempt to make a change in the first employee listed to replace Steve Stedman with Ben Franklin, one of the founding fathers and early presidents of the United States. If you are in another country and you are feeling patriotic, feel free to replace Ben Franklin with your own local historic figure.

To start with, lets first take a look at the data we are going to be changing:

```
 -- modifying data, first check the row we are going to change.  SELECT J
SON_VALUE(JsonData, '$.employees[0].firstName') as FirstName,         JSO
N_VALUE(JsonData, '$.employees[0].lastName') as LastName,   *   FROM XmlA
ndJson   WHERE ID = 1;  SELECT XmlData.value('(/employees/employee/firstN
ame)[1]', 'varchar(max)') as FirstName,     XmlData.value('(/employees/em
ployee/lastName)[1]', 'varchar(max)') as LastName,     *   FROM XmlAndJso
n   WHERE ID = 1;
```

We are going to do 4 things in this demo for both xml and json. Change the first name of the first entry to be Ben, the last name for that entry to Franklin, add a age of 341, and a hobby of kites. Yes Ben was a fan of flying kites. For each of these we are going to pull the data value out of the table, make several changes, then update it in the table. We could use an update statement to modify things every step of the way if that was needed.

Modifying JSON

```
--JSON_MODIFY DECLARE @json as NVARCHAR(MAX);  SELECT @json = JsonData
FROM XmlAndJson  WHERE ID = 1;  -- change firstname SET @json = JSON_MODI
FY(@json, '$.employees[0].firstName', 'Ben') ;  -- change lastname SET @j
son = JSON_MODIFY(@json, '$.employees[0].lastName', 'Franklin') ;  -- add
 age  SET @json = JSON_MODIFY(@json, '$.employees[0].age', '314') ;  -- a
dd hobby SET @json = JSON_MODIFY(@json, '$.employees[0].hobby', 'Kites')
;  SELECT @json;  UPDATE XmlAndJson    SET JsonData = @json  WHERE ID = 1
;
```

Once complete we will check our results.

```
SELECT JSON_VALUE(JsonData, '$.employees[0].firstName') as FirstName,
    JSON_VALUE(JsonData, '$.employees[0].lastName') as LastName,        J
SON_VALUE(JsonData, '$.employees[0].age') as Age,        JSON_VALUE(JsonD
ata, '$.employees[0].hobby') as Hobby,     *   FROM XmlAndJson  WHERE ID
= 1;
```

The JSON_MODIFY function was used to both change the existing values as well as to add new values.

Modifying XML

Modifying XML is a tiny bit more complex in that there are different options for updating an existing value versus inserting a new value. It is a slightly different syntax.

```
--XML Modify DECLARE @xml as XML;  SELECT @xml = XmlData   FROM XmlAndJso
n  WHERE ID = 1;    -- change firstname SET @xml.modify('replace value of
 (/employees/employee/firstName/text())[1] with "Ben"');  -- change lastn
ame SET @xml.modify('replace value of (/employees/employee/lastName/text(
))[1] with "Franklin"');  -- add age  SET @xml.modify('insert 314 into (/
employees/employee)[1]');  -- add hobby SET @xml.modify('insert Kites int
o (/employees/employee)[1]');  SELECT @xml;  UPDATE XmlAndJson    SET Xml
Data = @xml  WHERE ID = 1;
```

Then we check our changes:

```
 SELECT XmlData.value('(/employees/employee/firstName)[1]', 'varchar(max)
') as FirstName,   XmlData.value('(/employees/employee/lastName)[1]', 'va
rchar(max)') as LastName,   XmlData.value('(/employees/employee/age)[1]',
 'varchar(max)') as Age,   XmlData.value('(/employees/employee/hobby)[1]'
, 'varchar(max)') as Hobby,   *   FROM XmlAndJson   WHERE ID = 1;
```

Summary

You can see with either XML or JSON it is easy to make simple updates and changes to the data in either format. Perhaps the most complex part was figuring out the syntax to insert and update the XML values. Hopefully these examples will help with that.

Related Posts

- Getting started with XML and JSON in SQL Server.
- JSON and XML Month at SteveStedman.com
- SQL 2019 Page and Row Compression with XML and JSON
- Accessing XML and JSON content
- Iterating over JSON and XML values