Accessing XML and JSON content

Building on my earlier post of JSON and XML data, using the same table today we will take a look at accessing data.

With that we can easily grab the entire XML document or the entire JSON object, but what if we want to query a specific users first name or last name.

Here is how we would do that in JSON, using the JSON_VALUE function:

```
SELECT *,        JSON_VALUE(JsonData, '$.employees[0].firstName') as Firs
tName,        JSON_VALUE(JsonData, '$.employees[0].lastName') as LastName
   FROM XmlAndJson   WHERE ID = 1;
```

And here is how to do it with XML uising the .value function:

```
SELECT *,  XmlData.value('(/employees/employee/firstName)[1]', 'varchar(m
ax)') as FirstName,  XmlData.value('(/employees/employee/lastName)[1]', '
varchar(max)') as LastName   FROM XmlAndJson   WHERE ID = 1;
```

One thing to keep in mind is that the path is case sensitive for both XML and JSON. If you are getting NULL values for you output you may want to check the case on the paths.

Performance

So how do these compare on performance. Lets take a look by going through all 1,000,000 rows of the table and calling the functions to access the XML and to access the JSON. I initially tried it with the 10,000 rows from the previous example and the timing was pretty close. I wanted to do more rows for a better comparison.

To start with I hacked up the original queries above, to only return the first and last name, and to do it for all the rows in the table.

```
SET STATISTICS TIME ON;   SELECT --*,        JSON_VALUE(JsonData, '$.empl
oyees[0].firstName') as FirstName,        JSON_VALUE(JsonData, '$.employe
es[0].lastName') as LastName   FROM XmlAndJson   --WHERE ID = 1;    SELEC
T --*,  XmlData.value('(/employees/employee/firstName)[1]', 'varchar(max)
') as FirstName,  XmlData.value('(/employees/employee/lastName)[1]', 'var
char(max)') as LastName   FROM XmlAndJson   --WHERE ID = 1;  SET STATISTI
CS TIME OFF;
```

Using the STATISTICS TIME option we can see the following:

You can see that the elapsed time for JSON is 4.18 seconds and for XML it is 11.367 seconds. The XML takes 2.7 times longer to run that the JSON does in this example. If you are only accessing a couple rows that may not be a big deal, but if you are accessing thousands or millions of rows that may have a big impact on the overall performance.

Related Links

- 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