DB2 and XML

By,
Srikar Reddy Gondesi,
Junior SQL Server DBA,
Database Team,
Miracle Software Systems. Inc,
Email: sgondesi@miraclesoft.com

MIRACLE MSS
SOFTWARE SYSTEMS, INC.

Bird's Eye view of XML

- XML is a text-based markup language that is fast becoming the standard for data interchange on the web.
- As with HTML, you identify data using tags (identifiers enclosed in angle brackets: <...>). Collectively, the tags are known as markup.
- But unlike HTML, XML tags identify the data rather than specify how to display it. Whereas an HTML tag says something like, "Display this data in bold font" (...), an XML tag acts like a field name in your program.
- It puts a label on a piece of data that identifies it (for example, <message>...</message>).

Example:

Here is an example of some XML data you might use for a messaging application:

```
<message>
  <to>you@yourAddress.com</to>
  <from>me@myAddress.com</from>
  <subject>XML Is Really Cool</subject>
  <text>
    How many ways is XML cool? Let me count the ways...
  </text>
</message>
```

Create a DB2 table that can store XML data

- Create a table named MYCUSTOMER that contains an XML column:

CREATE TABLE MYCUSTOMER (Cid BIGINT, INFO XML);

MIRACLE MSS
SOFTWARE SYSTEMS, INC.

Create an index over XML data

- Assume that all XML documents that you store in the INFO column have a root element named customerinfo with an attribute named Cid. Create a unique index on the Cid attribute:

```
CREATE UNIQUE INDEX MYCUT_CID_XMLIDX ON
MYCUSTOMER(INFO)
 GENERATE KEY USING XMLPATTERN
 'declare default element namespace "http://posample.org";
/customerinfo/@Cid'
 AS SQL DECFLOAT
```

- The XML pattern that defines the index is case-sensitive. The element and attribute names in the XML pattern must match the element and attribute names in the XML documents exactly.

MIRACLE MSS
SOFTWARE SYSTEMS, INC.

Insert XML documents into XML column INFO

- Insert three XML documents into the MYCUSTOMER table

```
INSERT INTO MYCUSTOMER (CID, INFO) VALUES (1000,
'<customerinfo xmlns="http://posample.org" Cid="1000">
  <name>Kathy Smith</name>
  <addr country="Canada">
    <street>5 Rosewood</street>
    <city>Toronto</city>
    <prov-state>Ontario</prov-state>
    <pcode-zip>M6W 1E6</pcode-zip>
  </addr>
  <phone type="work">416-555-1358</phone>
</customerinfo>')
```

MIRACLE MSS
SOFTWARE SYSTEMS, INC.

Update XML documents that are stored in an XML column

- Update the document for which the CID column value is 1002, to add a cell phone number. The only way to change individual items in an XML column is to replace the entire column.

```
UPDATE MYCUSTOMER SET INFO =
'<customerinfo xmlns="http://posample.org" Cid="1002">
  <name>Jim Noodle</name> <addr country="Canada">
   <street>25 EastCreek</street>  <city>Markham</city>
   <prov-state>Ontario</prov-state>  <pcode-zip>N9C
    3T6</pcode-zip>
  </addr> <phone type="work">905-555-7258</phone>
  <phone type="cell">905-554-7254</phone> </customerinfo>'
WHERE CID=1002
```

MIRACLE MSS
SOFTWARE SYSTEMS, INC.

Delete rows based on the content of XML documents

- Delete any rows from the MYCUSTOMER table for which the customer document in the INFO column contains cell phone number. Issue the XMLEXISTS predicate with an XQuery expression to specify the documents that you want to delete.

```
DELETE FROM MYCUSTOMER
WHERE XMLEXISTS (
'declare default element namespace "http://posample.org";
 /customerinfo/phone[@type="cell"]' PASSING INFO)
```

MIRACLE MSS
SOFTWARE SYSTEMS, INC.

Query XML data

- Retrieve an entire XML document: Use a SELECT statement to retrieve the entire XML document that has a CID value of 1000.

```
SELECT CID, INFO FROM MYCUSTOMER
 WHERE CID=1000
```

MIRACLE MSS
SOFTWARE SYSTEMS, INC.

- Retrieve a portion of an XML document:

Use a SELECT statement with the XMLQUERY function to retrieve the name element from each XML document in the MYCUSTOMER table. Start of change

```
SELECT XMLQUERY (
  'declare default element namespace "http://posample.org";
   for $d in $doc/customerinfo
   return <out>{$d/name}</out>' passing INFO as "doc")
 FROM MYCUSTOMER as c WHERE XMLEXISTS ('declare
    default element namespace "http://posample.org";
   $i/customerinfo/addr[city="Toronto"]' passing c.INFO as "i")
```

MIRACLE MSS
SOFTWARE SYSTEMS, INC.

Update part of an XML document

- Change the address from 5 Rosewood to 42 Rosedale for the customer with customer ID 1000 in the MYCUSTOMER table. You issue the XMLMODIFY function to do that.

```
UPDATE MYCUSTOMER
 SET INFO = XMLMODIFY(
  'declare default element namespace "http://posample.org";
  replace value of node /customerinfo/addr/street
  with "42 Rosedale"')
 WHERE CID=1000#
```

MIRACLE MSS
SOFTWARE SYSTEMS, INC.

Add an XML type modifier

- Add an XML type modifier to the INFO column in the MYCUSTOMER table so that all documents that you store in the INFO column are automatically validated.

- Suppose that XML schema CU1 can be used to validate the XML data in the INFO column, and that XML schema CU1 is registered in the XML schema repository. Issue the following statement to add an XML type modifier to the INFO column that validates all data in the column against XML schema CU1.

```
ALTER TABLE MYCUSTOMER
 ALTER COLUMN INFO
 SET DATA TYPE XML (XMLSCHEMA ID SYSXSR.CU1)
```

MIRACLE MSS
SOFTWARE SYSTEMS, INC.

THANK YOU

MIRACLE MSS
SOFTWARE SYSTEMS, INC.