DB2 9 & XML

The first ‘pure’ implementation

Software Belux Technical Symposium
November 14, 2006

Stefan Van den Borre,
IT specialist, Database & Integration Services
+32.2.655.55.88 – +32.486.64.21.56
Stefan.vandenborre@be.ibm.com
XML conquers the world!!

Figure VI.2: Market Size by XML Data Store Solution Type

Market Size by XML Data Store Solution Type

- Purpose-Built
- XML Standalone
- General Purpose

Years:
- 1999
- 2000
- 2001
- 2002
- 2003
- 2004
- 2005
- 2006
- 2007
Once there were two worlds …

The Relational world

SQL

DB2

The XML world

XQuery / XPath

Lots of data is in relational DBs

That will not change…because SQL DBs are fast

XML as interchange: Web Services everywhere

XML as the transaction artifact: More and more data is represented as XML from the start…

All this is data

→ How do we join them?

New FIXML Protocol: extensible and lower application development & maintenance cost compared with old FIX protocol
1. Force XML into a relational schema

- The Relational world
  - SQL
  - DB2

- The XML world
  - XQuery / XPath

- **We leave XML World !!!!!**
  - Mapping from XML to relational often too complex
  - Often requires dozens or hundreds of tables
  - Complex multi-way joins to reconstruct documents
  - XML schema changes break the mapping, no schema flexibility!

For example: Change element from single-to multi-occurrence requires normalization of relational schema & data
2. Store XML as CLOB / Varchar

The Relational world

SQL

The XML world

XQuery / XPath

SQL knowledge needed
Query evaluation & sub-document level access requires costly XML Parsing = too slow!
And then Viper (*) came ...

The world changed...
At last a universal translator brought peace between the two people, once divided by their languages and storage characteristics.

From now on data could be stored the way it fitted best without any restrictions!!

(*) Viper was the code name for DB2 9
Some basics: What’s XML

✓ XML is vendor and platform independent
✓ Flexibility, Flexibility, Flexibility!
  • Any platform, vendor, OS, software, language
✓ XML is a very flexible data model: for structured data, semi-structured data, schema-less data
✓ Easy to extend: define new tags as needed
✓ XML is self-describing: any XML parser can "understand" it!
✓ Easy to "validate" XML, i.e. to check compliance with a schema - any XML parser can do it!
✓ Easy to transform XML documents into other formats (HTML, etc.)
✓ Fully Unicode compliant

<book>
  <authors>
    <author id="47">John Doe</author>
    <author id="58">Peter Pan</author>
  </authors>
  <title>Database systems</title>
  <price>29</price>
  <keywords>
    <keyword>SQL</keyword>
    <keyword>relational</keyword>
  </keywords>
</book>

XML = eXtensible Markup Language

XML is a self-describing "Lingua franca" for information exchange between organizations, applications, services, processes, ...
Some basics: XML Parsing, the Document Tree

```
<dept>
  <employee id=901>
    <name>John Doe</name>
    <phone>408 555 1212</phone>
    <office>344</office>
  </employee>
  <employee id=902>
    <name>Peter Pan</name>
    <phone>408 555 9918</phone>
    <office>216</office>
  </employee>
</dept>
```

XML text is represented as a document tree
(This is the XML standard and NOT specific to DB2 Viper)

Parse
DB2’s ‘Pure’ XML Storage

- **DB2 stores XML in parsed hierarchical format (~DOM)**

- **Example:**
  ```sql
  CREATE TABLE dept (deptID CHAR(8), ..., deptdoc XML);
  ```

  ```xml
  <dept> ...
  <emp>...<emp>
  </dept>
  ...
  ... ...
  ...
  ```

  **Relational columns are stored in relational format (tables)**

  **XML columns are stored natively**

  **No XML parsing for query evaluation!**
Efficient Document Tree Storage in Viper

SYSIBM.SYSXMLSTRINGS

String table

<table>
<thead>
<tr>
<th>0</th>
<th>dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>employee</td>
</tr>
<tr>
<td>1</td>
<td>name</td>
</tr>
<tr>
<td>5</td>
<td>id</td>
</tr>
<tr>
<td>2</td>
<td>phone</td>
</tr>
<tr>
<td>3</td>
<td>office</td>
</tr>
</tbody>
</table>

Tag names encoded as unique integers

Database must be in UTF-8
(Unicode is part of XML standard)

"Compression"

- Reduced storage
- Fast comparisons & navigation
- Maximum Document size = 2 GB (compressed)
Some basics: XQuery, the FLWOR Expression

XQuery = ‘XML SQL’

- Interrogate xml documents according the hierarchical nature of XML
- XQuery is a standard query language, not tied to DB2 Viper
- XQuery will thus not be covered further in this presentation

FOR: iterates through a sequence, bind variable to items
LET: binds a variable to a sequence
WHERE: eliminates items of the iteration
ORDER: reorders items of the iteration
RETURN: constructs query results

Example

create table dept(deptID char(8), deptdoc xml);

xquery
for $d in db2-fn:xmlcolumn('dept.deptdoc')/dept
let $emp := $d//employee/name
where $d/@bldg > 95
order by $d/@bldg
return
  <EmpList>
  {$d/@bldg, $emp}
  </EmpList>
However, SQL or XQuery, it doesn’t matter …

- **Storage data & Accessing data are fully decoupled**
  - SQL can be used to access data (XML elements) in XML columns
  - XQuery can be used to access relational tables data

- **Applications can combine XML & relational data access**
  - SQL or XQuery can be used independently from actual storage (see above)
  - SQL statement can contain XQuery expressions
  - XQuery statements can contain SQL expressions

Actually both SQL and XQuery are translated to a DB2 specific intermediate language, which is then optimize considering the way the data is stored (relational tables or pure XML)
### SQL/XML example

#### Dept

<table>
<thead>
<tr>
<th>deptID</th>
<th>unitID</th>
<th>deptdoc</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>110</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>220</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>240</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>310</td>
<td>30</td>
<td></td>
</tr>
</tbody>
</table>

#### Unit

<table>
<thead>
<tr>
<th>unitID</th>
<th>Empcount</th>
<th>bldg</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>234</td>
<td>G</td>
</tr>
<tr>
<td>20</td>
<td>123</td>
<td>H</td>
</tr>
<tr>
<td>30</td>
<td>32</td>
<td>H</td>
</tr>
</tbody>
</table>

#### XML Example

```xml
<dept bldg="G">
  <name>Engineering</name>
  <manager>Anjul</manager>
  <backup>Susan</backup>
  <admin>Mary</admin>
  <chargecode>CW345</chargecode>
</dept>
```

#### SQL with embedded XQuery

```sql
select d.deptID, xmlquery('__$deptdoc/dept/name' passing d.deptdoc as "deptdoc"), u.empcount
from dept d, unit u
where d.unitID = u.unitID and u.empcount > 200
and xmlquery('__$deptdoc/dept/@bldg' passing d.deptdoc as "deptdoc") = u.bldg
and xmlexists('__$deptdoc/dept/name' passing d.deptdoc as "deptdoc")
```

**SQL with embedded XQuery**
XML Indexes for High Query Performance

✓ Define 0, 1 or multiple XML Value Indexes per XML column
✓ Index any elements or attributes, incl. mixed content
✓ Index definition uses an XML pattern to specify which elements/attributes to index (and which not to)
✓ Can index all elements/attributes, but not forced to do so
✓ Can index repeating elements ⇒ 0, 1 or multiple index entries per document
✓ New XML-specific join and query evaluation methods, evaluate multiple predicates concurrently with minimal index I/O

create index xmlindex1 on dept(deptdoc) generate key using xmlpattern '/dept/name' as varchar(30);

<table>
<thead>
<tr>
<th>Tablename</th>
<th>XML Column</th>
<th>XML Pattern (index key)</th>
<th>SQL Conversion type</th>
</tr>
</thead>
</table>
XML Schema Repository (XSR)

- Also DTDs and External entities can be registered
  - Used for entity reference resolution and defaults
  - NOT used for validation
- Also used by Shred (see next slide)
  - Stores annotated Schema
  - Internal formats to make Shredding efficient

REGISTER XMLSCHEMA 'http://myPOschema/PO' FROM 'file://c:/TEMP/PO.xsd' AS user1.POschema

XML Schema Validation

Schema URI
Content URI
SQL 2-part name
Shredding into relational tables

✔ There are still reasons to shred XML.
  - Co-existence with legacy applications
  - Relational processing is faster than XML
  - Analytics/cubes work over non-XML data

✔ Mapping from XML to relational:
  - Annotate the XML schema
  - Register XML schemas in the schema repository
  - Shred via CLP commands or stored procedure calls

Annotated Schemas = XML Style Shredding
Replaces XML Extender shred (XML collection)

```xml
<xsd:element name="phone" type="xsd:string" db2-xdb:rowSet="employee_tab" db2-xdb:column="phone_col"/>
```

```
REGISTER XMLSCHEMA 'http://myEmployeeSchema/employee.xsd'
    FROM 'file://c:/temp/employee.xsd' AS john.myschema COMPLETE
    ENABLE DECOMPOSITION

DECOMPOSE XML DOCUMENT c:\mydoc.xml XMLSCHEMA john.myschema
```
Demo

SilverCastles demo (4'15")

A PHP coding example with DB2 Viper
The Viper way, pure XML in DB2

- **Flexibility because that is what XML is all about.**
  - Any document, Any schema, Not just the ones that are mapped to relational tables

- **XML storage: several options:**
  - Pure: XML is parsed and stored hierarchical.
  - Shredded: using annotated Schema (= replaces XML Extender)
  - BLOB

- **Sophisticated XML indexing**
  - "XANDOR join" to evaluate many predicates concurrently

- **Broad XQuery support**
  - Both embedded in SQL and as a primary language

- **Supports Digital Signatures**
  - Signatures can be validated on retrieved documents
DB2 9 & XML

Q & A …

Software Belux Technical Symposium
November 14, 2006

Stefan Van den Borre,
IT specialist, Database & Integration Services
+32.2.655.55.88 – +32.486.64.21.56
Stefan.vandenborre@be.ibm.com
XML

✓ Get off to a fast start with DB2 Viper (http://www-128.ibm.com/developerworks/db2/library/techarticle/dm-0603saracco/)

✓ Query DB2 XML Data with SQL (http://www-128.ibm.com/developerworks/db2/library/techarticle/dm-0603saracco2/)


✓ From DAD (= XML Extender) to annotated XML schema decomposition (http://www-128.ibm.com/developerworks/db2/library/techarticle/dm-0604pradhan)