Integration von XML Technologie in die Datenbank DB2

Manfred Päßler, IBM Technical Sales Datamanagement
manfred.paessler@de.ibm.com

IBM Software Group

Agenda

- How XML is used
- Tagging via SQL/XML
- Tagging via table tagging function
- Using XML Extender
- DB2 and Web Services
- Interesting internet adresses
How is XML used?

- Intra and inter company communications
- Browser based front-end

1. Transformations
2. Archiving/Searching
3. Visualisation

Tagging via SQL/XML

- **XMLELEMENT** function:
  constructs an XML Element out of an Element Expression and/or a list of Attributes

- **XMLATTRIBUTE** function:
  constructs XML attributes (within XMLELEMENT)

- **XMLAGG** function:
  returns the concatenation of a set of XML data (grouping/ordering with ORDER BY and sort expressions)

- **XML2CLOB**:
  returns an XML expression (XMLELEMENT and XMLAGG) as a CLOB.
The SQL/XML Background

SQL/XML: 2nd half of 2000, ANSI H2 and ISO SC32/WG3 approved new project for new part of SQL: “Part 14, XML-Related Specifications (SQL/XML)”

Official H2.3 task group (formally the informal SQLX workgroup):
- Mission: explore technologies and develop proposals for SQL extensions for XML
- Website: http://www.sqlx.org
- Members: Oracle, IBM, Microsoft, Sybase, FileTech,...

ISO/IEC JTC1 SC32 WG3

Tagging via SQL/XML – Examples

```
SELECT e.empno, XML2CLOB (XMLELEMENT (NAME "Emp", e.firstname || " " || e.lastname))
AS "Result"
FROM employee e
WHERE e.edlevel = 12

DEPTNO Result
A00 <Mgr ID="000010"/>
B01 <Mgr ID="000020"/>
C01 <Mgr ID="000030"/>
D01 <Mgr/>
```

```
SELECT d.deptno, XML2CLOB (XMLELEMENT (NAME "Mgr", XMLATTRIBUTES (d.mgrno)))
AS "Result"
FROM department d
WHERE d.admrdept = 'A00'
```
Tagging via SQL/XML – Examples

```
SELECT XML2CLOB(XMLELEMENT(NAME "Department",
    XMLATTRIBUTES(e.workdept AS "name"),
    XMLAGG(XMLELEMENT(NAME "emp",e.lastname)
        ORDER BY e.lastname))AS "dept_list"
FROM employee e
WHERE e.workdept IN ('C01','E21')
GROUP BY workdept
```

```
<row>
    <column name="DEPTNO">D01</column>
    <column name="MGRNO"null="true"/>
    <column name="ADMRDEPT">A00</column>
</row>
```

The where clause could again be an select expression
Using XML Extender

Client

- DB2
- CAE
- XML Extender Client
- Admin GUI
- Admin Command

Server

- DB2 Server
- XML Extender Server
- Stored procedures
- UDFs
- XML Parser

- UNIX (AIX, Linux, Solaris) and Windows
  - Incorporates ICU, XML4C and XML4J
  - Included in DB2 7+
- zSeries
  - Requires the XML toolkit 1.2 (later 1.3)
  - Included in DB2 V7+
- iSeries
  - XMLC included in the operating system
  - Runs on OS/400 5.1+

DB2 Extender Concepts

- **LOB TYPES**
  - Storing objects of any kind up to 2 GB into DB2
  - Data movement and externalisation only if needed (LOB locators)
- **USER-DEFINED FUNCTIONS**
  - Build SQL Functions in any language
  - Controlled by DB2
  - Overloading functions managed by DB2
  - Integrated into DB2 Optimizer
- **DISTINCT TYPES**
  - Create new data types for type safety on DB2 base types
  - Type dollar (=integer) not compatible with type DM (=integer)
  - Automatic generated casting functions
- **TRIGGERS**
  - Call function by event (event = SQL insert/update/delete)
  - Before/after triggers
- **Stored Procedures**
  - DB function calls to implement complex database logic

SQL Interface!

```
select txt-c1, img-c2, xml-c3
from mm-table
where
  fulltextsearch(txt-c1)
  qbic(img-c2)
  xpath(xml-c3)
```
How DB2 XML Extender supports XML

- Storing XML Docs
  - as is (into a column, external file or message queue) [UDFs]
  - decomposition into relational [Stored Procedures]
- Validate against DTD or X-Schema
- Extract/update attribute/element values from XML docs via XPath Syntax [UDFs]
- DB2 index crucial XML attribute/element values (Side Tables)
  - Fast DB2 search
  - combine XML docs with legacy relational data
- Fast fulltext search in XML structures [Textextender/Text Information Extender]
- Retrieve XML docs [UDFs]
- Composite/Decompositie XML docs from/to relational [Stored Procedures]
- Store/retrieve/extract/update/composite/decompose into/from message queue [UDFs+MQSeries]
- Transform XML docs via XSLT
- DTD Repository
- Administration Wizard [Java Program]

DB2 XML Extender -
Two Access and Storage Methods

- DAD (Data Access Definition)
- XML DOC
- DB2 XML Extender
- user table
- compose/ decompose
- collection
- side tables
- DTD repository
- MQ
- External Files
Using XML Columns

**Document centric approach**
- Store entire XML documents as column data in user tables
  - Documents already exist, exist externally
  - Documents primarily read-only, update performance not critical
  - Structural text search with section support
  - Documents with large text blocks
- DAD can create side tables for fast search of selected XML element/attribute values
- Provide UDF and UDT for defining and accessing content

**Data centric approach**
- Compose & decompose XML documents from and to data in relational tables
  - Data already stored in database
  - Document content update performance is important
  - Rationalize different document sources
- Application specific mapping using DAD
- Can create different documents ("XML views") from same data
- Dynamic XML queries can override DAD
- Stored procedures for composition and decomposition

Composition/Decomposition

```xml
<order key='99'>
  <customer>Thompson</customer>
  <part key='82'> .... </part>
  <part key='83'> .... </part>
</order>
```

<table>
<thead>
<tr>
<th>order_tab</th>
<th>part_tab</th>
</tr>
</thead>
<tbody>
<tr>
<td>order_key</td>
<td>part_key</td>
</tr>
<tr>
<td>99</td>
<td>82</td>
</tr>
<tr>
<td>Thompson</td>
<td>83</td>
</tr>
</tbody>
</table>

Mapping Methods:
- SQL (composition only)
- RDB_node (XML-Style for composition/decomposition)
Fast Search with Side Tables

Store UDF

User Table

Side Tables

DAD
(Data Access Definition)

DAD (Data Access Definition)

define Indexes

XPath

XColumn Definition in DAD

<?xml version="1.0"?>
<!DOCTYPE DAD SYSTEM "c:\dxx\std\dad.dtd">
<DAD>
<dtdid>getstart.dtd</dtdid>
<validation>YES</validation>
<xcolumns>
<table name="order_side_tab">
<column name="order_key" type="integer" path="/Order/@key" multi_occurrence="NO"/>
<column name="customer" type="varchar(50)" path="/Order/Customer/Name" multi_occurrence="NO"/>
</table>
<table name="part_side_tab">
<column name="price" type="decimal(10,2)" path="/Order/Part/ExtendedPrice" multi_occurrence="YES"/>
</table>
<table name="ship_side_tab">
<column name="date" type="DATE" path="/Order/Part/Shipment/ShipDate" multi_occurrence="YES"/>
</table>
</xcolumns>
</DAD>
XML Transformations

1. DAD Creation
   - DB2 XML Wizard, DAD Checker
   - IBM WebSphere Studio Application Developer
   - application ... on the fly ....
   - can include XSL Stylesheet (in advance or on the fly)

2. Tooling
   - DB2 XML Administration Wizard
     - GUI based front-end for enabling columns and collections
     - WebSphere Studio
     - An advanced development environment for J2EE application development
     - Includes comprehensive XML tooling including
       - XML and DTD editors
       - RDB to Database Mapping tool (aka DAD Builder)
WSAD: XML Tooling Features

- **DTD/Schema Editor**
  - Visual tooling for working with DTDs/Schemas
  - Create DTDs/Schemas from existing XML documents
  - Convert to/from DTD or Schema
  - Generate JavaBean(s) for creating/manipulating XML documents from DTD/Schema
  - Generate an HTML form from a DTD

- **XML Source Editor**
  - Design/Source mode
  - DTD/Schema validation
  - Code Assist for building XML documents

WSAD: XML Tooling Features

- **XML Mapping Editor**
  - Generate XSL to map XML between DTDs/Schemas

- **XML to/from Relational Databases**
  - Generate XML, XSL, XSD from an SQL Query

- **RDB/XML Mapping Editor**
  - Map columns in a table to elements/attributes to XML
  - Generate a Database Access Definition (DAD) script to compose/decompose XML documents to/from a database
  - DAD is used with DB2 XML Extender
Using XML UDFs

- Stores a record into an XML-Table with UDF:
  SQL INSERT INTO sales (ID, NAME, ORDER)
  VALUES ("1234", "Sriram" XMLCLOBFromFile("c:/dxx/samples/cmd/order.xml"))

- Exports XML documents as varchars into files
  select db2xml.Content(db2xml.xmlvarchar(order), order.xml') from sales where ID=1234

- Selects attributes and values from stored XML documents
  SELECT extractVarchar(Order, '/Order/Customer/Name') from sales
  WHERE id > 10

- Updates an XML document with UDF
  UPDATE sales
  set order = Update(order, '/Order/Customer/Name', 'Heine')
  WHERE name = 'Sriram'

- Fulltext search in XML-Doc with Text Information Extender
  SELECT Order FROM Sales
  WHERE Contains(section(/Order/Customer/Name), "Heine") = 1

---

Extracting multible Values UDF Example

- Select * from table(db2xml.extractIntegers(
  db2xml.XMLFile("c:/dxx/samples/xml/book1.xml"), '/book/*/@id')) as x;

```xml
<book><chapter id="1"> </...>
<footnote id="2">...</...
```
Using XML Collection Stored Procedures

**Decomposition**
- `dxxInsertXML(collection_name,XML_Doc,...)`
- `dxxShredXML(DAD,XML_Doc,...)`

**Composition**
- `dxxGenXML(DAD,result_table,max_rows,condition,..)`
- `dxxRetrieveXML(collection_name,.....)`

**Composition into CLOB**
- `dxxRetrieveXMLClob(Collection_name,...,CLOB)`
- `dxxGenXMLClob(DAD,...,CLOB,..)`

Using XML Extender and MQSeries

- A series of SQL User Defined Functions and Stored Procedures are supplied in DB2 7.2 enabling a DB2 client application to do the following from a single request:
  - To read from MQSeries queues into DB2 XML columns
  - To send and publish messages to MQSeries queues from DB2 XML columns
  - To send XML messages composed from relational data to MQSeries queues
  - To decompose (shred) XML messages held in MQSeries queues into relational data

- It is possible to integrate MQSeries operations with DB2 table operations in a single SQL request, such as a SELECT
Understanding Web Services

- Week Integration on Application Web

Understanding Web Services continued

- Improved Integration on Application Web
Understand Web Services continued

The new Web Service Economy

- Complex Network of Application to Application Interaction

Web Services Business Model

- **XML** defines a universal way of representing any data, making data integration simple
- **SOAP** uses XML as messages to define a universal Web service requests, making process integration simple
- **WSDL** specifies all information needed for integration, making universal application assembly tools possible
- **UDDI** is a special Web service which allows users and applications to locate required Web services
- **WS-I.org** formed to achieve seamless interoperability
Integration Advantages with Web Services

- Support different programming languages
- APIs can change
- Different operating systems or hardware platforms
- Different software vendors, in-house code
- Loose coupling with SOAP over HTTP (no client software, easy firewall setup)

Invoke a Web Service with DB2 access

SQL
Stored Procedures
XML Extender
Business example

- Company BlueAirways has Stored Procedures that implement a certain business logic
  - Retrieve all flights from city1 to city2 on a certain date
- BlueAirways wants to share this with company TravelPortal to build a travel web portal
  - BlueAirways creates web service definition file for SPs, cities and dates being parameters of the web service
  - Deploys Web Service and sends WSDL to TravelPortal
  - TravelPortal creates web application using BlueAirways’s web service
  - Invocation of DB2 WS Provider will execute the SP and return the result as XML
  - Web Service can be shared with other companies too

Defining Web Services

- Web Services are defined through simple XML files
  - Users list database operations to be exposed as Web Services
- Provider Runtime will help
  - Deploy Web Service
  - Generate WSDL
  - Provide HTML test environment to run Web Service
  - Automatic regeneration if Web Service definition changes
- Deployment
  - WebSphere (including zSeries and iSeries) and Apache Tomcat

```xml
<DADX>
<operation name="listDepartments">  
  <documentation>Example Employee web service</documentation>
  <query>
    <SQL_query>
      SELECT * FROM DEPARTMENT WHERE LOCATION = :location
    </SQL_query>
    <parameter name="location" type="xsd:string"/>
  </query>
</operation>
</DADX>
```
Development Scenario

DB2 as a Web Service Consumer

- Deploys UDFs for implementing an "SQL SOAP proxy"
- Provides tools for generating DB2 table functions related to WSDL definition (WSAD)
- Usage: invoke Web Services as an SQL function

```
SELECT name, symbol, STOCK_QUOTE(symbol) AS quote FROM stock_watch
```
**Development Scenario**

- **DB2 Client**
  - Soap Proxy
  - Find WSDL

- **WEB Service Provider**
  - Publish WSDL

- **DB2 Developer**
- **UDDI Registry**
  - Find WSDL

**Tooling**

- WebSphere Studio allows end to end development of DB-oriented Web Services
  - Create SQL queries with SQL editor or XML Extender DADs
  - Create Web Service definition files (DADX)
  - Create DB2 Web Service Table Functions from WSDL
  - Deploys Web Services to WebSphere and Apache Tomcat
  - Publishes WSDL in UDDI
  - Test of Web Service
  - Generation of Java Web Service clients
Interesting Internet Addresses

• DB2 XML Extender Homepage

• Information Integration Technology
  preview for information integration via federation and XQuery

• alphaWorks
  www.alphaworks.ibm.com - web site for free emerging technologies from IBM.

• developerWorks
  www.ibm.com/developer/xml - web site for product- and platform-independent information
  on e-business application development

• e-business
  www.ibm.com/e-business - site for more information on IBM e-business products

• WebSphere
  http://www-4.ibm.com/software/webservers/