Incorporating Alternative Data Structures

Chapter 13 (XML) with Added Material

Incorporating Alternative Data Structures

- The need to store complex data types has increased sharply in the last decade (maybe two).
- NoSQL databases typically use complex data types (e.g. document databases).
- In this topic, we explore tow of the most popular formatting methods:
  - XML (Extensible Markup Language)
  - JSON (JavaScript Object Notation)

XML

- XML (Extensible Markup Language) is:
  - A general-purpose markup language
  - Used to describe documents in a format that is convenient for display on web pages
  - Available in many platforms, including standard SQL (SQL/XML), which was added in 2003 and subsequently revised.

XML Documents

- XML is best understood as a standard framework for creating document markup languages
- There are now over 100 standard document formats built using XML, including:
  - RSS
  - Atom
  - SOAP
  - XHTML
  - Office Open XML
  - HR-XML

XML Basics

- ***Markup Language***: a set of annotations, often called ***tags***, used to describe how text is structured, formatted, or laid out.
- XML syntax is similar to HTML
  - Both are based on the Standard Generalized Markup Language (SGML)
  - Which, in turn, is based on the Generalized Markup Language (GML) developed by IBM in the 1960s
- Tagged text is intended to be human-readable
- HTML has predefined tags; in XML, you defined your own

Example Using SQL Query

PERFORMERS

PERF_ID: INT	PERF_NAME: VARCHAR(60)
3003	Stevie Wonder
3010	Pink Floyd

CD_INVENTORY

CD_NAME: VARCHAR(60)	PERF_ID: INT	IN_STOCK: INT
Innervisions (Remastered)	3003	16
The Definitive Collection	3003	34
Dark Side of the Moon	3010	27
The Endless River	3010	12

```
SELECT a.PERF_NAME, a.PERF_ID, b.CD_NAME, b.IN_STOCK
  FROM PERFORMERS a
  JOIN CD_INVENTORY b
    ON a.PERF_ID = b.PERF_ID
 ORDER BY a.PERF_NAME, b.CD_NAME;
```

Results in Text

PERF_ID	PERF_NAME	CD_NAME	IN_STOCK
3010	Pink Floyd	Dark Side of the Moon	27
3010	Pink Floyd	The Endless River	12
3003	Stevie Wonder	Innervisions (Remastered)	16
3003	Stevie Wonder	The Definitive Collection	34

- Suitable for display or printing
- Not as suited to passing to another application or embedding in HTML for a web page

Results in XML

```
<artists>
  <artist id="3010">
    <name>Pink Floyd</name>
      <CDs>
        <CD stock="27"><name>Dark Side of the Moon</name></CD>
        <CD stock="12"><name>The Endless River</name></CD>
      </CDs>
  </artist>
  <artist id="3003">
    <name>Stevie Wonder</name>
      <CDs>
        <CD stock="16"><name>Innervisions (Remastered)</name></CD>
        <CD stock="34"><name>The Definitive Collection</name></CD>
      </CDs>
  </artist>
  <!-- Additional artists available soon -->
</artists>
```

Basic Syntax: Tags

- Tags
  - Formed using names enclosed in angle brackets
  - Each tag has a matching end tag (same name prefixed by a slash.
    - Example: <artists> has an end tag of </artists>
  - By Custom (and best practice):
    - Lists have plural names
    - Single items have singular names
  - Comments have tag names beginning with <!--
    - Example: `<!-- This is a comment -->`

Basic Syntax: Data Items and Values

- Data Items and Values
  - Coded as name-value pairs in one of two ways:
    - XML ***attribute***
      - Name the value inside another tag
      - Name followed by equal sign and data value enclosed by double-quote characters.
      - Example: `<artist id="3003">`
    - XML ***element***
      - Separate tag for data element
      - Data value sandwiched between start and end tags
      - Example: `<name>Pink Floyd</name>`
  - Guideline: Use element when item might be broken down further (perhaps at a later time)

SQL and XML

- Unlike SQL, XML can show the data's hierarchy
  - Collections of tree structures called a ***forest***
- Indentation is a best practice, but not required
  - White space between tags is ignored
- XML coding can be tedious
  - However, there are tools for encoding/decoding
  - For SQL, many implementations have functions for handling XML (SQL/XML functions appear later in this topic)
- NoSQL Document and Key-Value implementations can also use XML (or JSON)

SQL/XML (in Standard SQL)

- Two important components:
  - XML data type
  - SQL/XML functions
- We now look at both of these…

XML Data Type

- XML Data Type in SQL
  - Handled the same general way as other SQL data types
  - A simple and logical extension
  - Not the only way to use XML and SQL together
    - Alternatively, could store XML in VARCHAR or CLOB type column
  - Advantage is that SQL engine can provide features tailored to handling XML structures
  - Available in
    - Oracle and DB2 UDB support the SQL standard implementation
    - SQL Server supports XML, but not per the SQL standard

Specifying Column as XML

- Defining Column as XML Type:
  - General specification format:
    - `XML( <type modifier> {( <secondary type modifier> )} )`
    - Type modifier required, enclosed in parentheses
      - **DOCUMENT**: text document formatted in XML
      - **CONTENT**: includes complex data, possibly including binary data
      - **SEQUENCE**: XQuery document (out of scope in this course)
    - Secondary type modifier is optional:
      - **UNTYPED**: not of a particular type
      - **ANY**: any type supported by the SQL implementation
      - **XMLSCHEMA**: registered XML schema, such as:
        - **Xs**: The W3C XML Schema
        - **sqlxml**: ISO sqlxml schema

SQL DDL Including XML Type

```
CREATE TABLE ARTISTS
( ARTIST_ID          INT,
  ARTIST_NAME        VARCHAR(60),
  ARTIST_DOB         DATE,
  POSTER_IN_STOCK    BOOLEAN,
  ARTIST_BIOGRAPHY XML(DOCUMENT(UNTYPED)) );
```

SQL/XML Functions

- SQL/XML Function:
  - Like any other SQL function (returns a single value with each execution)
  - However, value returned is formatted as XML

Standard XML Functions (1)

Function	Value Returned
XMLAGG	A single XML value containing an XML forest formed by combining (aggregating) a collection of rows that each contain a single XML value
XMLATTRIBUTES	One or more attributes in the form name=value within an XMLELEMENT
XMLCOMMENT	An XML comment
XMLCONCAT	A concatenated list of XML values, creating a single value containing an XML forest
XMLDOCUMENT	An XML value containing a single document node
XMLELEMENT	An XML element, which can be a child of a document node, with the name specified in the name parameter

Standard XML Functions (2)

Function	Value Returned
XMLFOREST	An XML element containing a sequence of XML elements formed from table columns, using the name of each column as the corresponding element name
XMLPARSE	An XML value formed by parsing the supplied string without validating it
XMLPI	An XML value containing an XML processing instruction
XMLQUERY	The result of an XQuery expression (out of scope)
XMLTEXT	An XML value containing a single XML text node (can be a child of an XML document)
XMLVALIDATE	An XML sequence that is the result of validating and XML value

SQL/XML Query

- Query using SQL/XML Functions:

```
SELECT XMLELEMENT(NAME "ArtistCD",
       XMLFOREST(a.PERF_NAME as Artist, a.PERF_ID,
       b.CD_NAME, b.IN_STOCK)
  FROM PERFORMERS a
  JOIN CD_INVENTORY b
       ON a.PERF_ID = b.PERF_ID
 WHERE a.PERF_ID = '3003'
   AND a.PERF_ID = b.PERF_ID
 ORDER BY b.CD_NAME;
```

- XML element name will be taken from column name in the result set. Use column alias to customize (e.g. AS Artist)

Query Results in XML

```
<ArtistCD>
  <Artist>Stevie Wonder</Artist>
  <PERF_ID>3003</PERF_ID>
  <CD_NAME>Innervisions (Remastered)</CD_NAME>
  <IN_STOCK>16</IN_STOCK>
</ArtistCD>
<ArtistCD>
  <Artist>Stevie Wonder</Artist>
  <PERF_ID>3003</PERF_ID>
  <CD_NAME>The Definitive Collection</CD_NAME>
  <IN_STOCK>34</IN_STOCK>
</ArtistCD>
```

Incorporating JSON Data

- JSON (JavaScript Object Notation):
  - Is an open-standard format
  - Uses human-readable text for storing and transmitting data objects
    - Data objects consist of key-value (AKA name-value) pairs
  - Often preferred over XML because:
    - It is more compact
    - Structures are easier to traverse using programming languages, particularly scripting languages

JSON Data Types

Data Type	Description
Number	Signed decimal number that may contain a fractional part and may use exponential E notation
String	Sequence of zero or more Unicode characters, delimited by double-quotation marks
Boolean	Logical data type that permits only the values true and false
Array	Ordered list of zero or more values, each of which may be of any data type. Arrays are enclosed in square brackets, with commas separating the values
Object	Unordered collection of name-value pairs were the names (also known as keys) are strings. Objects are enclosed in curly brackets, with the name and value within each pair separated by a colon, and pairs with the object separated by commas
Null	Empty value, indicated by the word null

JSON Syntax Rules

- JSON has very simple syntax rules:
  - Whitespace (spaces, tabs, line feeds, and carriage returns) is ignored
  - No syntax for comments
  - Names and Values:
    - Each enclosed in double-quote characters
    - Separated by a colon
  - Arrays:
    - Enclosed in curly brackets: {}
    - Comma-separated list of elements

JSON Example

- Previous XML example formatted using JSON:

```
{
  "ArtistCD": {
    "Artist": "Stevie Wonder",
    "PERF_ID": "3003",
    "CD_NAME": "Innervisions (Remastered)",
    "IN_STOCK": "16"
  },
  "ArtistCD": {
    "Artist": "Stevie Wonder",
    "PERF_ID": "3003",
    "CD_NAME": "The Definitive Collection",
    "IN_STOCK": "34"
  }
}
```

JSON Implementation

- Commonly used for storage in NoSQL Document databases, such as Mongo
- Can also be use for values in a key-value store
- No specific support in SQL at this time
  - However VARCHAR and CLOB columns can be used to store data formatted as JSON documents