**Introduction to Semistructured Data and XML**

Chapter 27, Part D
Based on slides by Dan Suciu
University of Washington

---

**How the Web is Today**

- HTML documents
  - often generated by applications
  - consumed by humans only
  - easy access: across platforms, across organizations
- No application interoperability:
  - HTML not understood by applications
  - screen scraping brittle
  - Database technology: client-server
    - still vendor specific

---

**New Universal Data Exchange Format: XML**

A recommendation from the W3C
- XML = data
- XML generated by applications
- XML consumed by applications
- Easy access: across platforms, organizations
Paradigm Shift on the Web

- From documents (HTML) to data (XML)
- From information retrieval to data management
- For databases, also a paradigm shift:
  - from relational model to semistructured data
  - from data processing to data/query translation
  - from storage to transport

Semistructured Data

Origins:

- Integration of heterogeneous sources
- Data sources with non-rigid structure
  - Biological data
  - Web data

The Semistructured Data Model
Syntax for Semistructured Data

May omit oids:

```json
```

Characteristics of Semistructured Data

- Missing or additional attributes
- Multiple attributes
- Different types in different objects
- Heterogeneous collections

Self-describing, irregular data, no a priori structure
Comparison with Relational Data

<table>
<thead>
<tr>
<th>name</th>
<th>phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>3634</td>
</tr>
<tr>
<td>Sue</td>
<td>6343</td>
</tr>
<tr>
<td>Dick</td>
<td>6363</td>
</tr>
</tbody>
</table>

XML

- A W3C standard to complement HTML
- Origins: Structured text SGML
  - Large-scale electronic publishing
  - Data exchange on the web
- Motivation:
  - HTML describes presentation
  - XML describes content

From HTML to XML

Bibliography

Foundations of Databases, Abiteboul, Hull, Vianu
Addison-Wesley, 1995

Data on the Web, Abiteboul, Buneman, Suciu
Morgan Kaufmann, 1999

HTML describes the presentation
**HTML**

```html
<h1> Bibliography </h1>
<p> <i> Foundations of Databases </i>
    Abiteboul, Hull, Vianu
    <br> Addison Wesley, 1995
<p> <i> Data on the Web </i>
    Abiteboul, Buneman, Suciu
    <br> Morgan Kaufmann, 1999
```

**XML**

```xml
<bibliography>
    <book>
        <title> Foundations... </title>
        <author> Abiteboul </author>
        <author> Hull </author>
        <author> Vianu </author>
        <publisher> Addison Wesley </publisher>
        <year> 1995 </year>
    </book>
</bibliography>
```

**XML describes the content**

**Why are we DB’ers interested?**

- It’s data, stupid. That’s us.
- Proof by Google:
  - database+XML – 1,940,000 pages.
- Database issues:
  - How are we going to model XML? (graphs).
  - How are we going to query XML? (XQuery)
  - How are we going to store XML (in a relational database? object-oriented? native?)
  - How are we going to process XML efficiently? (many interesting research questions!)
**Document Type Descriptors**

- Sort of like a schema but not really.

```xml
<!ELEMENT Book (title, author*) >
<!ELEMENT title #PCDATA>
<!ELEMENT author (name, address, age?)>
<!ATTLIST Book id ID #REQUIRED>
<!ATTLIST Book pub IDREF #IMPLIED>
```

- Inherited from SGML DTD standard
- BNF grammar establishing constraints on element structure and content
- Definitions of entities

**Shortcomings of DTDs**

Useful for documents, but not so good for data:

- Element name and type are associated globally
- No support for structural re-use
  - Object-oriented-like structures aren’t supported
- No support for data types
  - Can’t do data validation
- Can have a single key item (ID), but:
  - No support for multi-attribute keys
  - No support for foreign keys (references to other keys)
  - No constraints on IDREFs (reference only a Section)

**XML Schema**

- In XML format
- Element names and types associated locally
- Includes primitive data types (integers, strings, dates, etc.)
- Supports value-based constraints (integers > 100)
- User-definable structured types
- Inheritance (extension or restriction)
- Foreign keys
- Element-type reference constraints
Sample XML Schema

```xml
<schema version="1.0" xmlns="http://www.w3.org/1999/XMLSchema">
  <element name="author" type="string" />  
  <element name="date" type = "date" />  
  <element name="abstract">
    <type>
      ...
    </type>
  </element>
  <element name="paper">
    <type>
      <attribute name="keywords" type="string"/>
      <element ref="author" minOccurs="0" maxOccurs="*" />  
      <element ref="date" />  
      <element ref="abstract" minOccurs="0" maxOccurs="1" />  
      <element ref="body" />  
    </type>
  </element>
</schema>
```

Important XML Standards

- XSL/XSLT: presentation and transformation standards
- RDF: resource description framework (meta-info such as ratings, categorizations, etc.)
- Xpath/Xpointer/Xlink: standard for linking to documents and elements within
- Namespaces: for resolving name clashes
- DOM: Document Object Model for manipulating XML documents
- SAX: Simple API for XML parsing
- XQuery: query language

XML Data Model (Graph)

Issues:
- Distinguish between attributes and sub-elements?
- Should we conserve order?
**XML Terminology**

- **Tags**: book, title, author, …
- **Elements**: `<book>`…`<book>`, `<author>`…`</author>`
  - elements can be nested
  - empty element: `<red/>` (Can be abbrev. `<red/>`)
- XML document: Has a single root element
- Well-formed XML document: Has matching tags
- Valid XML document: conforms to a schema

**More XML: Attributes**

```xml
<book price="55" currency="USD">
  <title>Foundations of Databases</title>
  <author>Abiteboul</author>
  ...
  <year>1995</year>
</book>
```

Attributes are alternative ways to represent data

**More XML: Oids and References**

```xml
<person id="o555"><name> Jane </name></person>

<person id="o456"><name> Mary </name>
  <children idref="o123 o555"/>
</person>

<person id="o123" mother="o456"><name>John</name></person>
```

oids and references in XML are just syntax
XML-Query Data Model

- Describes XML data as a tree
- Node ::= DocNode | ElemNode | ValueNode | AttrNode | NSNode | PINode | CommentNode | InfoItemNode | RefNode

http://www.w3.org/TR/query-datamodel/2/2001

XML-Query Data Model

Element node (simplified definition):

- elemNode : (QNameValue, {AttrNode}, [ElemNode | ValueNode]) → ElemNode
- QNameValue = means “a tag name”

Reads: “Give me a tag, a set of attributes, a list of elements/values, and I will return an element”

XML Query Data Model

Example:

```xml
<book price = "55" currency = "USD">
  <title> Foundations … </title>
  <author> Abiteboul </author>
  <author> Hull </author>
  <author> Vianu </author>
  <year> 1995 </year>
</book>
```

```xml
book1= elemNode(book, {price2, currency3}, [title4, author5, author6, author7, year8])
price2 = attrNode(…) /* next */
currency3 = attrNode(…)
title4 = elemNode(title, string9)
...```

```xml
<book price = "55" currency = "USD">
  <title> Foundations … </title>
  <author> Abiteboul </author>
  <author> Hull </author>
  <author> Vianu </author>
  <year> 1995 </year>
</book>
```
XML Query Data Model

Attribute node:

- attrNode : (QNameValue, ValueNode) → AttrNode

Example:

```xml
<book price = "55" currency = "USD">
  <title> Foundations … </title>
  <author> Abiteboul </author>
  <author> Hull </author>
  <author> Vianu </author>
  <year> 1995 </year>
</book>
```

```
price2 = attrNode(price,string10)
string10 = valueNode(…)

currency3 = attrNode(currency,string11)
string11 = valueNode(…)
```

XML Query Data Model

Value node:

- ValueNode = stringValue | boolValue | floatValue …

- stringValue : string → stringValue
- boolValue : boolean → boolValue
- floatValue : float → floatValue
XML Query Data Model

Example:

```xml
<book price = "55" currency = "USD">
  <title> Foundations … </title>
  <author> Abiteboul </author>
  <author> Hull </author>
  <author> Vianu </author>
  <year> 1995 </year>
</book>
```

```python
price2 = attrNode(price, stringValue("55"))
currency3 = attrNode(currency, stringValue("USD"))
title4 = elemNode(title, stringValue("Foundations…"))
```

XML vs. Semistructured Data

- Both described best by a graph
- Both are schema-less, self-describing
- XML is ordered, ssd is not
- XML can mix text and elements:
  ```xml
  <talk> Making Java easier to type and easier to type
    <speaker> Phil Wadler </speaker>
  </talk>
  ```
- XML has lots of other stuff: attributes, entities, processing instructions, comments