CSCC43H: Introduction to Databases

Lecture 10

Wael Aboulsaadat

Acknowledgment: these slides are partially based on Prof. Garcia-Molina & Prof. Ullman slides accompanying the course's textbook.

Database Management System (DBMS)

- A collection of programs that enable:

Defining (describing the structure)	DDL
Populating by data (Constructing)	DML
Manipulating (querying, updating)	DML
Preserving consistency	DDL, Normalization
– Protecting from misuse,	DDL, (security)
– Recovering from failure, and	Transactions
– Concurrent using	Transactions

of a database.

Introduction to XML

?? How Hot is XML?

- How many times “hotter” than “relational”?
  - A. 0.2 - 0.5
  - B. 0.5 - 1.0
  - C. 1.0 - 3.0
  - D. 3.0 - 10.0
  - E. more than 10.0

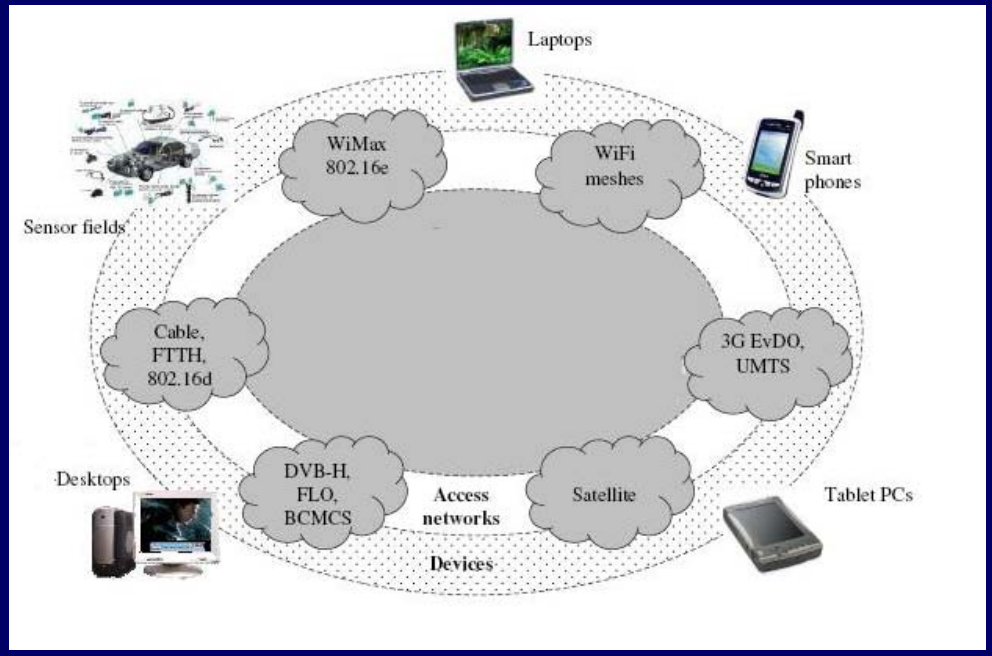
How Hot is XML: 10 to 70+ times Hotter

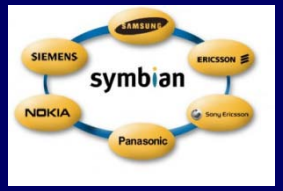
- How "hot" are they (based on 2003 logs….)
  - xml: 18.4 million hits at Google.com (was 10m, 2002)
    - 2002: 10m, 2001: 3.7m
  - relational: 1.48 million hits
    - 2002: 0.97m, 2001: 0.6m
- How many times "hotter"?
  - now: 18.4m/1.48m = 12.43 times
    - 2002: 10m/0.97m = 10.3 times
    - 2001: 3.7m/0.6m = 6.2 times
  - normalized by # years in existence: Stabilized? Going down?
    - 18.4/(2003-1998) vs. 1.48/(2003-1972) = 77.08 times
    - 2002: 10m/(2002-1998) vs. 0.97/(2002-1972) = 77.3
    - 2001: 3.7m/(2001-1998) vs. 0.6/(2001-1972) = 59.6

What is XML?

- *e***X***tensible* **M***arkup* **L***anguage*
- Markup language for documents containing structured information
- Defined by four specifications:
  - XML, the Extensible Markup Language
  - XLL, the Extensible Linking Language
  - XSL, the Extensible Style Language
  - XUA, the XML User Agent

How can we exchange data between heterogeneous systems?





Message in the Bottle (or: towards the Digital Rosetta Stone)

Degree of "self-description":

not quite

^@Some Quotations from the Universal Library^M1 Famous Quotes^M1.1 By William I^M[2, Sonnet XVIII]^MShall I compare thee to a summer's day?^MThou art more lovely and more temperate.^MRough winds do shake the darling buds of May,^MAnd summer's lease hath all too short a date.^MSometime too hot the eye of heaven shines,^MAnd often is his gold complexion dimmed.^MAnd every fair from fair some declines,^MBy chance or nature's changing course untrimmed.^MBut thy eternal summer shall not fade,^MNor lose possession of that fair thou owest,^MNor shall Death brag thou wander'st in his shade^MWhile in eternal lines to time thou growest.^MSo long as men can breathe, or eyes can see,^MSo long live this, and this gives life to thee.^M1.2 By William II^M[1, p.265]^M\223The obvious mathematical breakthrough would be development of^Man easy way to factor large prime numbers."^MReferences^M[1] W. H. Gates. The Road Ahead. Viking Penguin, 1995.^M[2] W. Shakespeare. The Sonnets of Shakespeare.609.^M^@

Two Important Ideas: (1) Markup?

- Information added to a text to make its structure comprehensible

- Pre-computer markup (punctuational and presentational)

Two Important Ideas: (2) declarative

- Names and structure
- Finer level of detail (most human-legible signals are overloaded)
- Independent of presentation (abstract)
- People often call this “semantic”

Message in the Bottle (or: towards the Digital Rosetta Stone)

Degree of "self-description"

not quite

```
^@Some Quotations from the Universal Library^M1
    Famous Quotes^M1.1 By William I^M[2,
    Sonnet XVIII]^MShall I compare thee to a
    summer's day?^MThou art more lovely and
    more temperate.^MRough winds do shake the
    darling buds of May,^MAnd summer's lease
    hath all too short a date.^MSometime too hot
    the eye of heaven shines,^MAnd often is his
    gold complexion dimmed.^MAnd every fair
    from fair some declines,^MBy chance or
    nature's changing course untrimmed.^MBut thy
    eternal summer shall not fade,^MNor lose
    possession of that fair thou owest,^MNor shall
    Death brag thou wander'st in his
    shade^MWhile in eternal lines to time thou
    growest.^MSo long as men can breathe, or
    eyes can see,^MSo long live this, and this
    gives life to thee.^M1.2 By William II^M[1,
    p.265]^M\223The obvious mathematical
    breakthrough would be development of^Man
    easy way to factor large prime
    numbers."^MReferences^M[1] W. H. Gates.
    The Road Ahead. Viking Penguin, 1995.^M[2]
    W. Shakespeare. The Sonnets of
    Shakespeare.609.^M^@^@^@^@^@^@^@^
    @^@^@^@^@^@^@^@^@^@^@^@^@^@
```

not bad

```
\documentclass{article}
\begin{document}
\title{Some Quotations from the
    Universal Library}
...
\section{Famous Quotes}
\subsection{By William I}
\textbf{\cite[Sonnet
    XVIII]{shakespeare-sonnets-
    1609}}
  \begin{verse}
    Shall I compare thee to a summer's
      day?\\
    Thou art more lovely and more
      temperate. \\
    Rough winds do shake the darling buds
      of May, \\
    And summer's lease hath all too short a
      date. \\
    Sometime too hot the eye of heaven
      shines, \\
    And often is his gold complexion
      dimmed. \\
  ...
    \qquad So long as men can breathe, or
      eyes can see,\\
    \qquad So long live this, and this gives
      life to thee.   \\
  \end{verse}
...
\bibliographystyle{abbrv}
\bibliography{msg}

\end{document}
```

XML: Basic format

1) Element: **<tag>**content**</tag>**

- basic unit
- tag name defines what the content is
- opening and closing tags enclose content

2) Attribute: Information about the data

- Attribute names are usually adjectives
- Stored as `attribute="value"` pairs:
  - **<**tag **attribute="value">**
  - content
  - </tag>

Rules for well-formed XML

- Elements that contain data must have <start> and </end> tags!
- Empty tags must be closed `<some-tag/>`
- Elements should not overlap

  Bad Nesting:

  `<trunk> <branch> </trunk> </branch>`
- All attribute values must be wrapped in quotes

  `<a href="newpage.html">`
- XML is case sensitive: <TAG> and <Tag> are treated differently. (Standard: use lower case.)

More XML Rules

- A document begins with:
  - an ***XML Declaration***

```
<?xml version="1.0" encoding="UTF-8"?>
```

  - and a ***DocType Declaration***:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML
   1.0 Strict//EN“
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
   strict.dtd">
```

- ***Root element*** immediately follows; encloses entire content of the document.

```
<book>
    everything else
</book>
```

Elements and their Content

element type

element

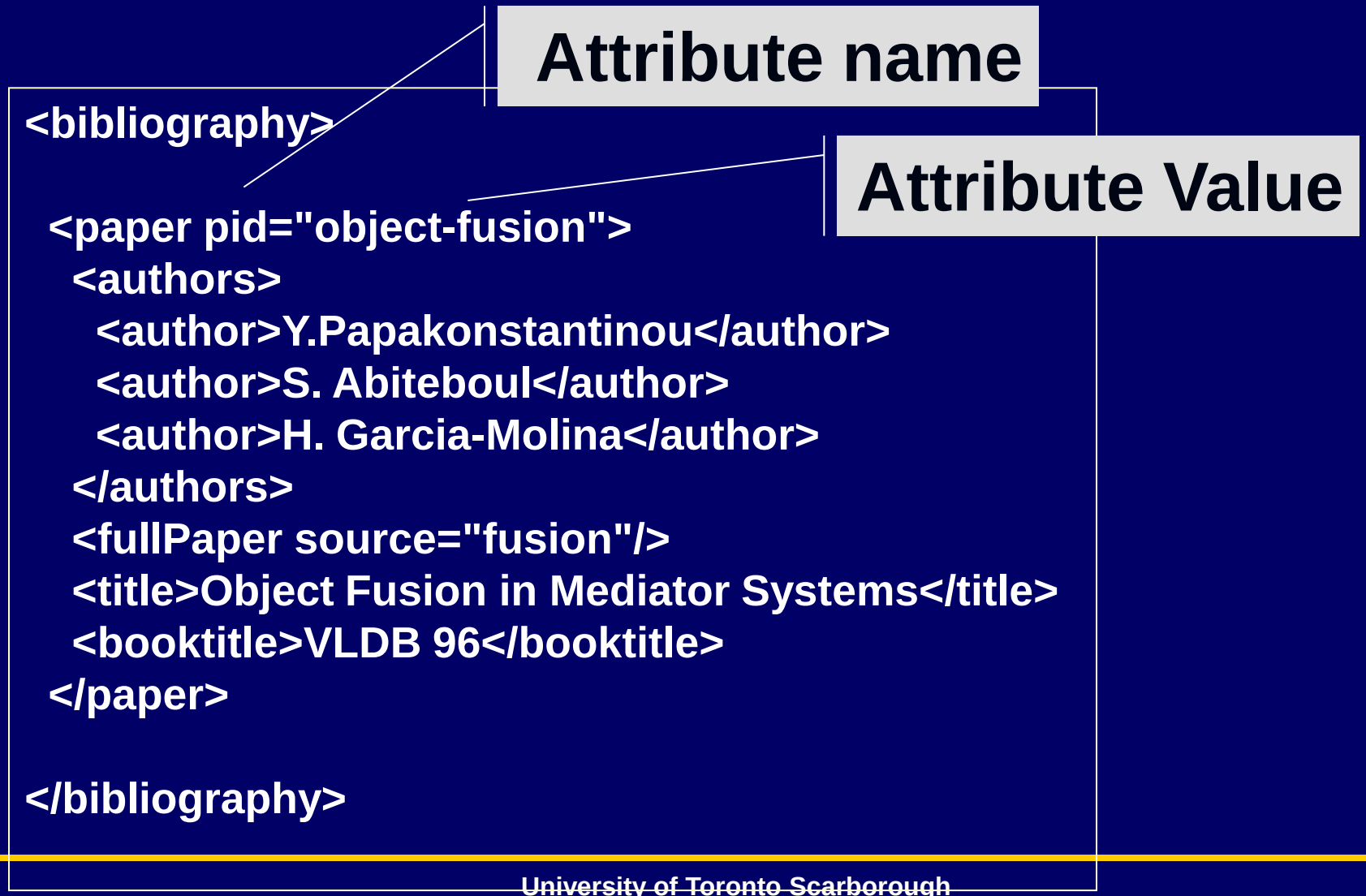
element content

empty element

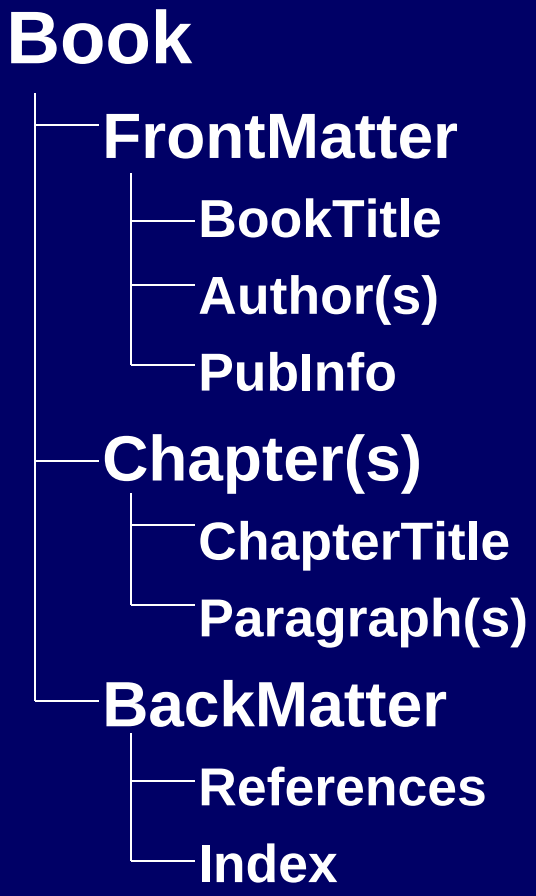
character content

```
<bibliography>
  <paper ID="object-fusion">
    <authors>
      <author>Y.Papakonstantinou</author>
      <author>S. Abiteboul</author>
      <author>H. Garcia-Molina</author>
    </authors>
    <fullPaper source="fusion"/>
    <title>Object Fusion in Mediator Systems</title>
    <booktitle>VLDB 96</booktitle>
  </paper>
</bibliography>
```

Element Attributes



XML Example: content objects in a book



A simple XML fragment

```
<Book>
  <FrontMatter>
    <BookTitle>XML Is Easy</BookTitle>
    <Author>Tim Cole</Author>
    <Author>Tom Habing</Author>
    <PubInfo>CDP Press, 2002</PubInfo>
  </FrontMatter>
  <Chapter>
    <ChapterTitle>First Was SGML</ChapterTitle>
    <Paragraph>Once upon a time …</Paragraph>
  </Chapter>
</Book>
```

This is NOT XML, why?

```
<PoemFragment>
  <Stanza>
    <Line><Sentence>It was six men of Indostan</Line>
    <Line>To learning much inclined,</Line>
    <Line>Who went to see the Elephant</Line>
    <Line>(Though all of them were blind),</Line>
    <Line>That each by observation</Line>
    <Line>Might satisfy his mind.</Sentence></Line>
  </Stanza>
</PoemFragment>
```

This is NOT XML, why?

```
<PoemFragment>
  <Stanza>
    <Line><Sentence>It was six men of Indostan</Line>
    <Line>To learning much inclined,</Line>
    <Line>Who went to see the Elephant</Line>
    <Line>(Though all of them were blind),</Line>
    <Line>That each by observation</Line>
    <Line>Might satisfy his mind </Sentence></Line>
  </Stanza>
</PoemFragment>
```

Message in the Bottle (or: towards the Digital Rosetta Stone)

Degree of "self-description":

not quite

```
^@Some Quotations from the Universal Library^M1
    Famous Quotes^M1.1 By William I^M[2,
    Sonnet XVIII]^MShall I compare thee to a
    summer's day?^MThou art more lovely and
    more temperate.^MRough winds do shake the
    darling buds of May,^MAnd summer's lease
    hath all too short a date.^MSometime too hot
    the eye of heaven shines,^MAnd often is his
    gold complexion dimmed.^MAnd every fair
    from fair some declines,^MBy chance or
    nature's changing course untrimmed.^MBut thy
    eternal summer shall not fade,^MNor lose
    possession of that fair thou owest,^MNor shall
    Death brag thou wander'st in his
    shade^MWhile in eternal lines to time thou
    growest.^MSo long as men can breathe, or
    eyes can see,^MSo long live this, and this
    gives life to thee.^M1.2 By William II^M[1,
    p.265]^M\223The obvious mathematical
    breakthrough would be development of^Man
    easy way to factor large prime
    numbers."^MReferences^M[1] W. H. Gates.
    The Road Ahead. Viking Penguin, 1995.^M[2]
    W. Shakespeare. The Sonnets of
    Shakespeare.609.^M^@^@^@^@^@^@^@^
    @^@^@^@^@^@^@^@^@^@^@^@^@^@^@
```

not bad

```
\documentclass{article}
 \begin{document}
 \title{Some Quotations from the
        Universal Library}
...
\section{Famous Quotes}
 \subsection{By William I}
 \textbf{\cite[Sonnet
        XVIII]{shakespeare-sonnets-
        1609}}
   \begin{verse}
    Shall I compare thee to a summer's
        day?\\
    Thou art more lovely and more
        temperate. \\
    Rough winds do shake the darling buds
        of May, \\
    And summer's lease hath all too short a
        date. \\
    Sometime too hot the eye of heaven
        shines, \\
    And often is his gold complexion
        dimmed. \\
   ...
    \qquad So long as men can breathe, or
        eyes can see,\\
    \qquad So long live this, and this gives
        life to thee.   \\
   \end{verse}
...
 \bibliographystyle{abbrv}
\bibliography{msg}

\end{document}
```

pretty good

```
<?xml version="1.0"?>
<universal_library>
 <books>
  <book> <title>Some Quotations from the Universal
      Library</title>
   <section> <title>Famous Quotes</title>
    <subsection>  <title>By William I</title>
     <quote bibref="shakespeare-sonnets-1609">
     <title>Sonnet XVIII</title>
     <verse>
      <line>Shall I compare thee to a summer's
     day?</line>
      <line>Thou art more lovely and more
     temperate. </line>
      <line>Rough winds do shake the darling buds of
     May, </line>
     </verse>
 ...
    <subsection> <title>By William II</title>
     <quote bibref="gates-road-ahead-1995">
      <title>Page 265</title>
      <line>``The obvious mathematical breakthrough
      would be development of an easy way to factor
      large prime numbers.”</line>
     </quote>
    </subsection>
   </section>
</book>
...
</books>
</universal_library>
```

XML Industry Initiatives

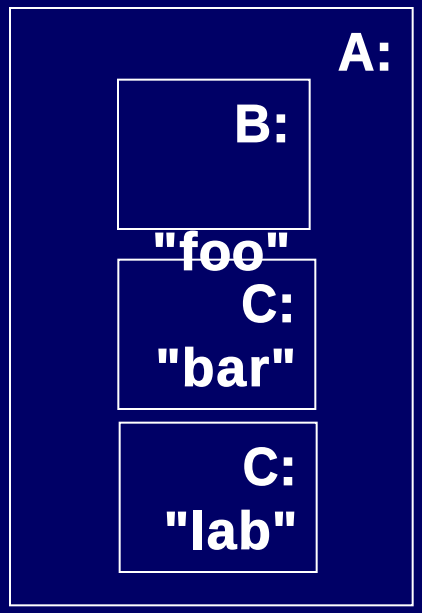
- Every community is building it's own XML protocols, e.g.:

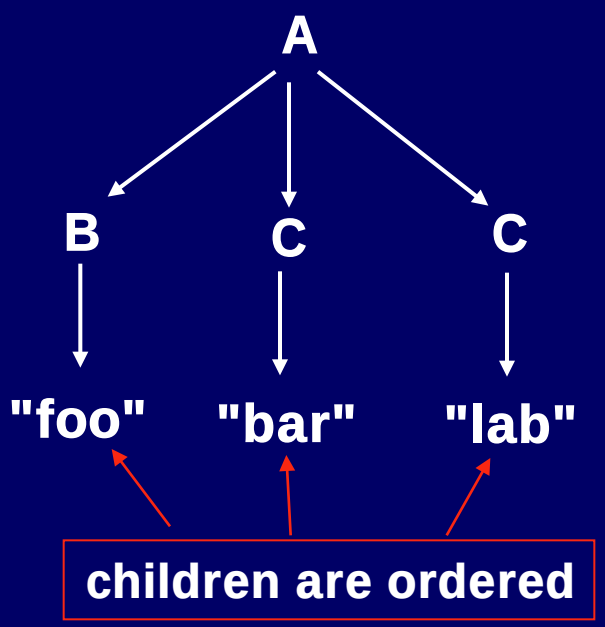
- Advertising: adXML place an ad onto an ad network or to a single vendor
- Literature: Gutenberg convert the world's great literature into XML
- Web Servers: apacheXML parsers, XSL, web publishing
- Travel: openTravel information for airlines, hotels, and car rental places
- News: NewsML creation, transfer and delivery of news
- Voice: VoxML markup language for voice applications
- Wireless: WAP (Wireless Application Protocol) wireless devices
- Weather: OMF Weather Observation Markup Format (simulation)
- Geospatial: ANZMETA distributed national directory for land information
- Banking: MBA Mortgage Bankers Association of America --> credit report, loan file, underwriting…
- Healthcare: HL7 DTDs for prescriptions, policies & procedures, clinical trials
- Math: MathML (Mathematical Markup Language)
- Surveys: DDI (Data Documentation Initiative) "codebooks" in the social and behavioral sciences

Http://www.oasis-open.org/cover/xml.html#applications

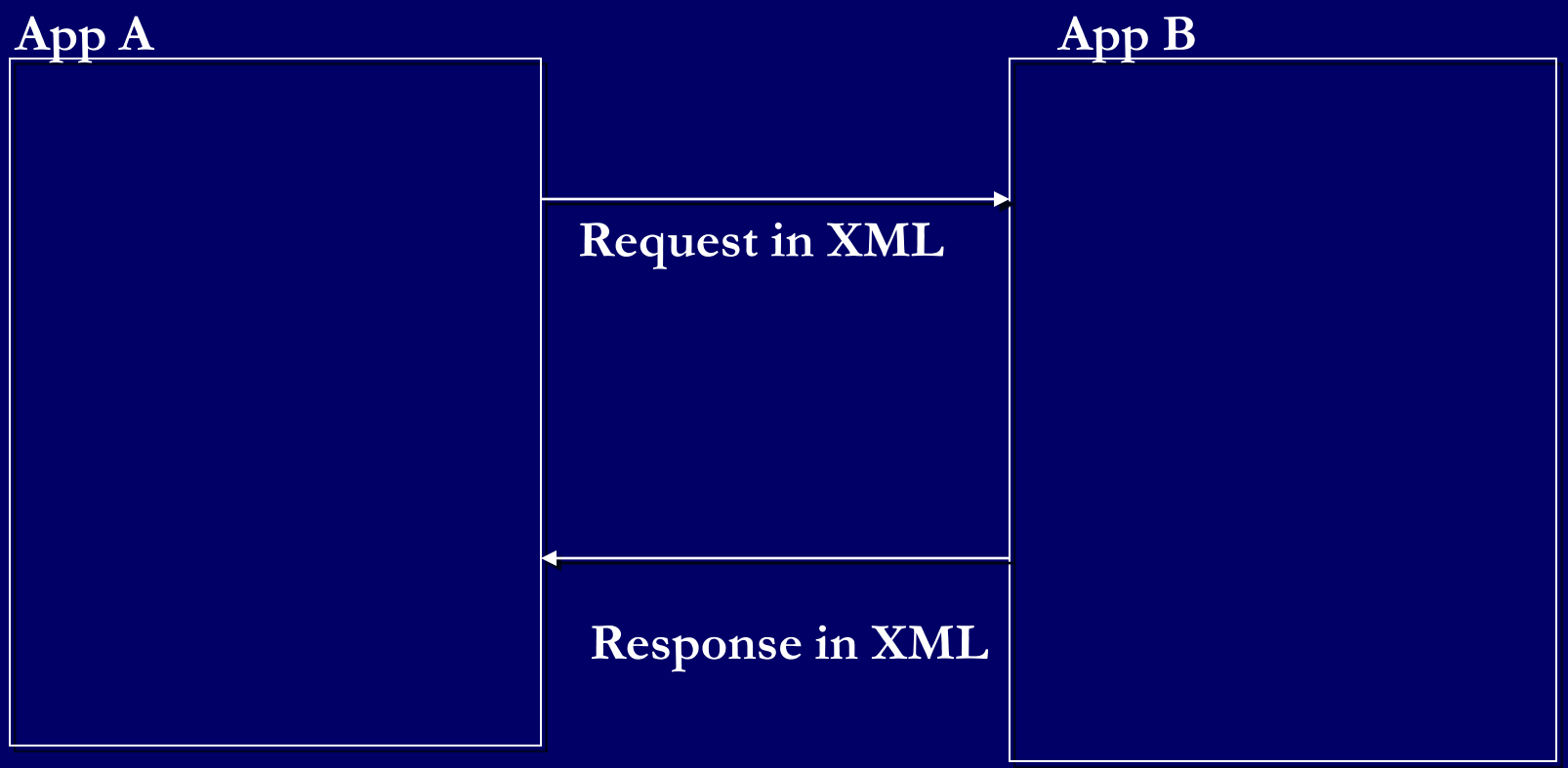
XML -- Instance Model

```
<A>
<B>foo</B>
<C>bar</C>
<C>lab</C>
</A>
```

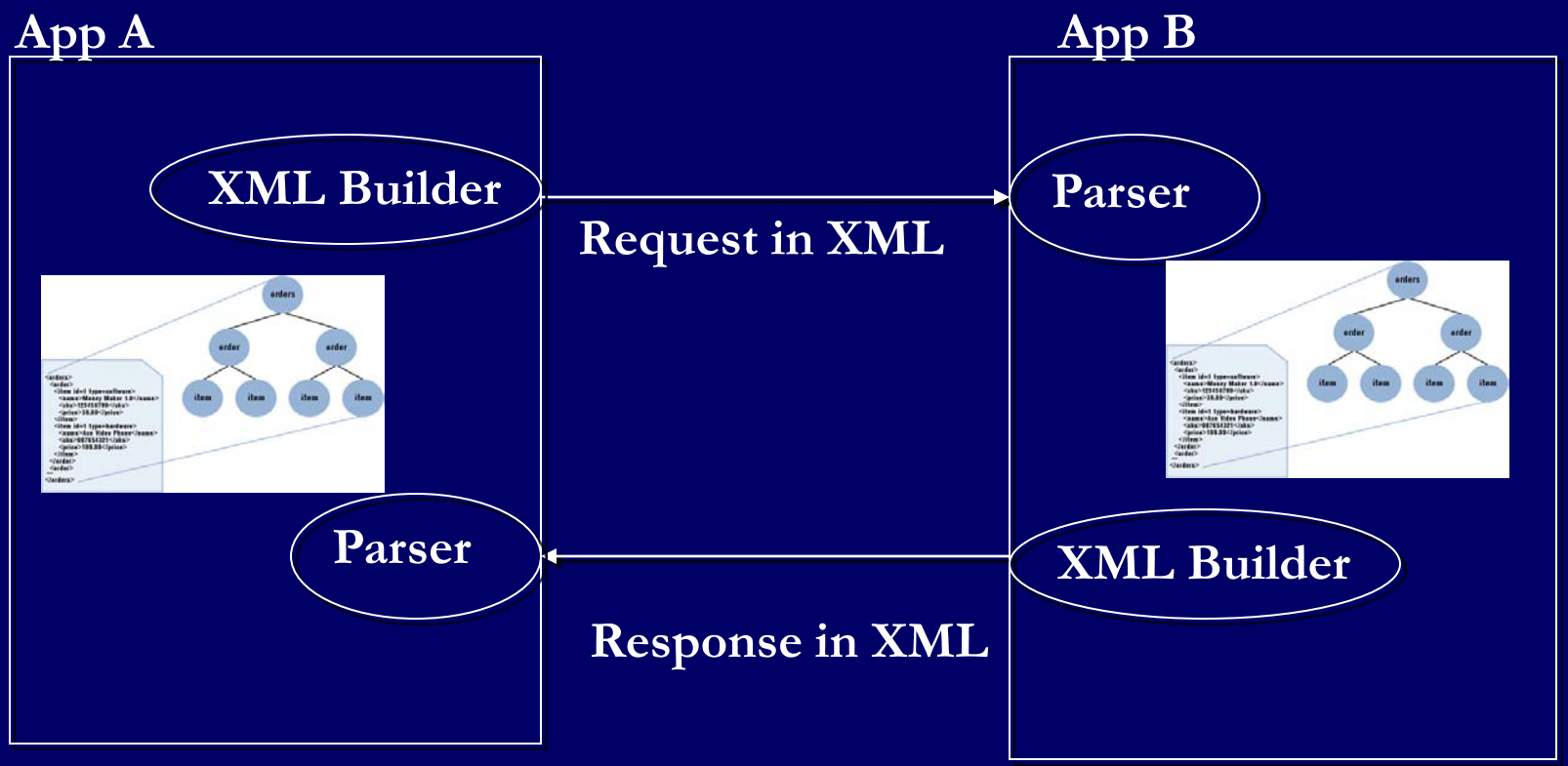




Two Applications Communicating

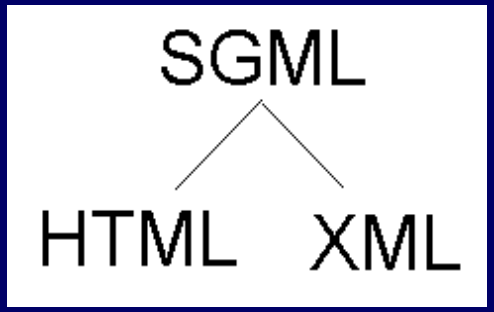


Two Applications Communicating



XML….

- Based on Standard Generalized Markup Language (SGML)
- Version 1.0 introduced by World Wide Web Consortium (W3C) in 1998
- Bridge for data exchange on the Web



Comparisons

XML

- Extensible set of tags
- Content orientated
- Standard Data infrastructure
- Allows multiple output forms

HTML

- Fixed set of tags
- Presentation oriented
- No data validation capabilities
- Single presentation

Authoring XML Elements

- An XML element is made up of a start tag, an end tag, and data in between.
- Example:

  <director> Matthew Dunn </director>
- Example of another element with the same value:

  <actor> Matthew Dunn </actor>
- XML tags are case-sensitive:

  <CITY> <City> <city>
- XML can abbreviate empty elements, for example:

  <married> </married> can be abbreviated to

  <married/>

Authoring XML Elements (cont'd)

- An attribute is a name-value pair separated by an equal sign (=).
- Example:

  <City ZIP="94608"> Emeryville </City>

- Attributes are used to attach additional, secondary information to an element.

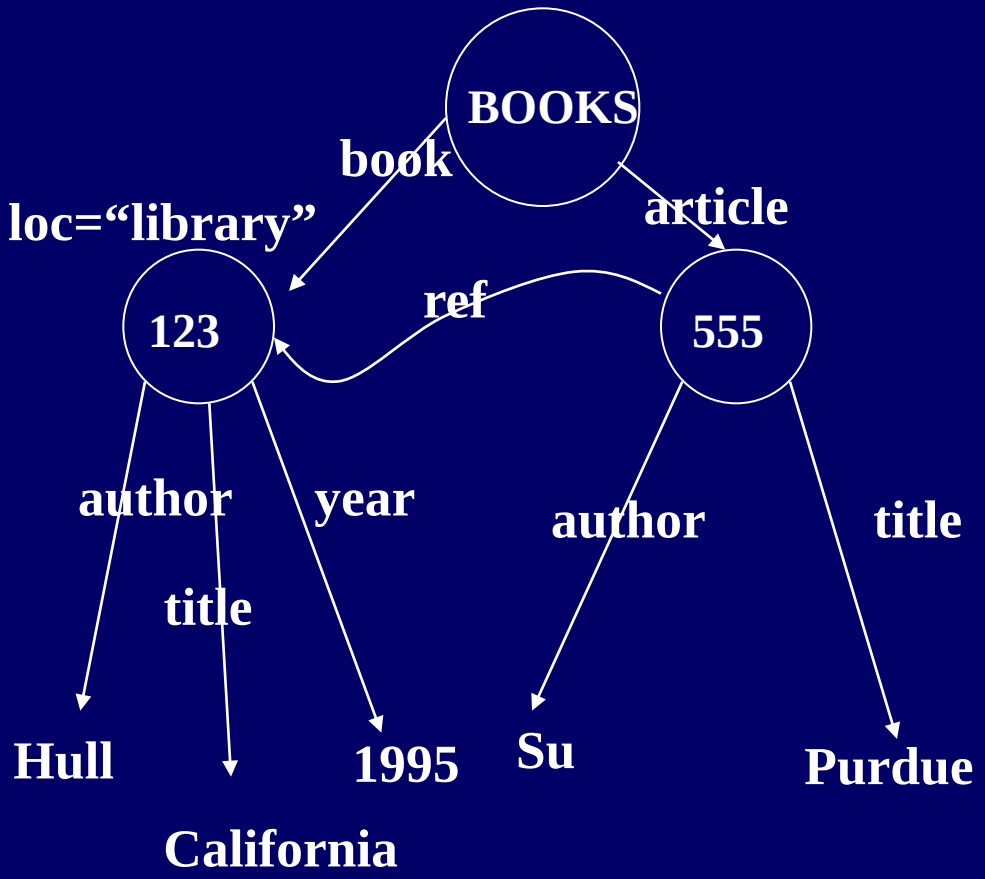
Authoring XML Documents

- A basic XML document is an XML element that can, but might not, include nested XML elements.
- Example:

```
<books>
    <book isbn="123">
        <title> Second Chance </title>
        <author> Matthew Dunn </author>
    </book>
</books>
```

XML Data Model: Example

```
<BOOKS>
<book id="123"
   loc="library">
  <author>Hull</author>
  <title>California</title>
  <year> 1995 </year>
</book>
<article id="555"
   ref="123">
  <author>Su</author>
  <title> Purdue</title>
</article>
</BOOKS>
```



Authoring XML Documents (cont'd)

- Authoring guidelines:
  - All elements must have an end tag.
  - All elements must be cleanly nested (overlapping elements are not allowed).
  - All attribute values must be enclosed in quotation marks.
  - Each document must have a unique first element, the root node.

Authoring XML Data Islands

- A data island is an XML document that exists within an HTML page.
- The <XML> element marks the beginning of the data island, and its ID attribute provides a name that you can use to reference the data island.

Authoring XML Data Islands (cont'd)

- Example:

```
<XML ID="XMLID">
    <customer>
        <name> Mark Hanson </name>
        <custID> 29085 </custID>
    </customer>
</XML>
```

Document Type Definitions (DTD)

- An XML document may have an optional DTD.
- DTD serves as grammar for the underlying XML document, and it is part of XML language.
- DTDs are somewhat unsatisfactory, but no consensus exists so far beyond the basic DTDs.
- DTD has the form:

  <!DOCTYPE name [markupdeclaration]>

DTD (cont'd)

- Consider an XML document:

```
<db><person><name>Alan</name>
                    <age>42</age>
                    <email>agb@usa.net </email>
        </person>
        <person>……</person>
        ……
</db>
```

DTD (cont'd)

- DTD for it might be:

```
<!DOCTYPE db [
    <!ELEMENT db (person*)>
    <!ELEMENT person (name, age, email)>
    <!ELEMENT name (#PCDATA)>
    <!ELEMENT age (#PCDATA)>
    <!ELEMENT email (#PCDATA)>
 ]>
```

DTD (cont'd)

Occurrence Indicator:

Indicator	Occurrence	
(no indicator)	Required	One and only one
?	Optional	None or one
*	Optional, repeatable	None, one, or more
+	Required, repeatable	One or more

XML and Databases

Three Approaches to Putting XML into a Database

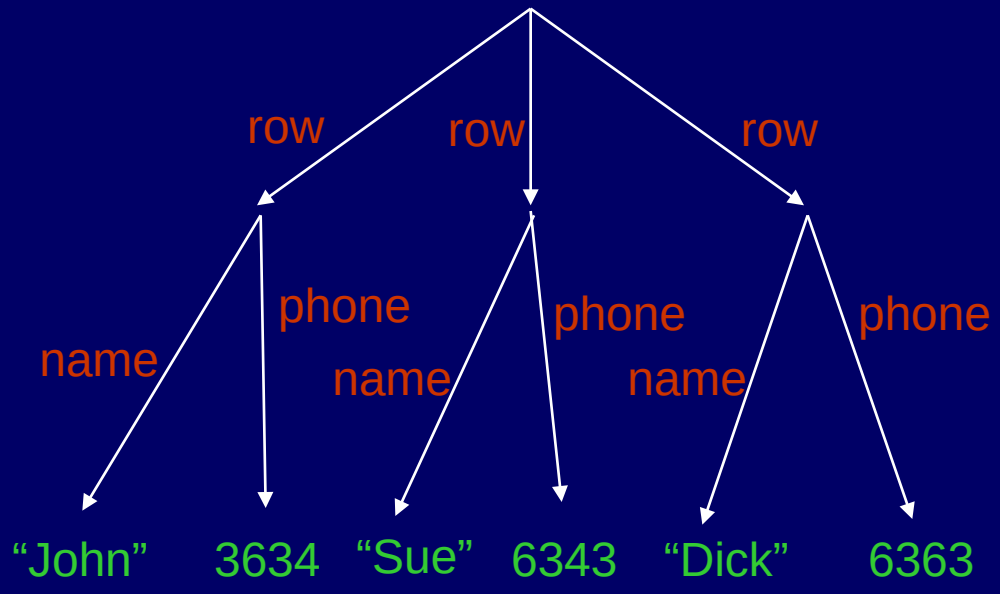
- One: Store XML document as a text BLOB
- Two: “XML Column”
  - Provide an XML object class for database values
    - An ORDBMS extension
  - Good for storing complete already existing documents
  - Good for archives (e.g. legal record)
  - Problematic when operating on or updating parts of documents (as opposed to whole document)

Three Approaches to Putting XML into a Database

- Three: “Shred and Publish” or “XML Collection”
  - Shred XML document content into a standard Relational Database
  - Publish query results as XML documents
  - Only works when
    - Documents conform strictly to DTDs or Schemas
    - There is no requirement to keep original document intact (e.g. store tags)
  - Good when parts of document are queried and updated
  - Issue: Mapping of Relations in database (a network) in XML tag structure (a hierarchy)

XML vs. Relational Data

name	phone
John	3634
Sue	6343
Dick	6363



Relation

XML

Query Language for XML

- Must be high-level; “SQL for XML”
- Must conform to DTD/XML Schema
  - But also work in absence of schema info
- Support simple and complex/nested datatypes
- Support universal and existential quantifiers, aggregation
- Operations on sequences and hierarchies of document structures
- Capability to transform and create XML structures

Overview of XQuery

- Path expressions (XPath)
- Element constructors
- FLWOR ("flower") expressions
  - Several other kinds of expressions as well, including conditional expressions, list expressions, quantified expressions, etc.
- Expressions evaluated w.r.t. a context:
  - Context item (current node)
  - Context position (in sequence being processed)
  - Context size (of the sequence being processed)
  - Context also includes namespaces, variables, functions, date, etc.

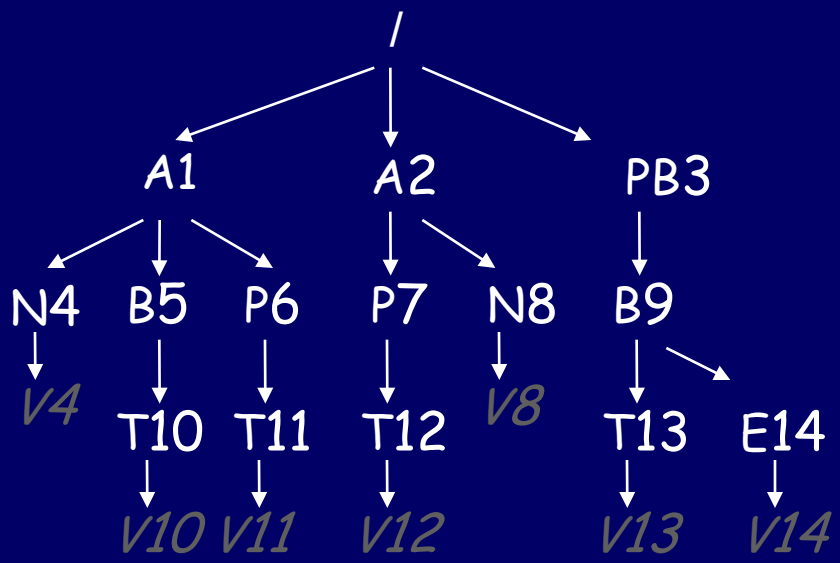
XPath Expressions

Examples:

- /booklist/book
- /booklist/book/author
- /booklist/book/author/lastname

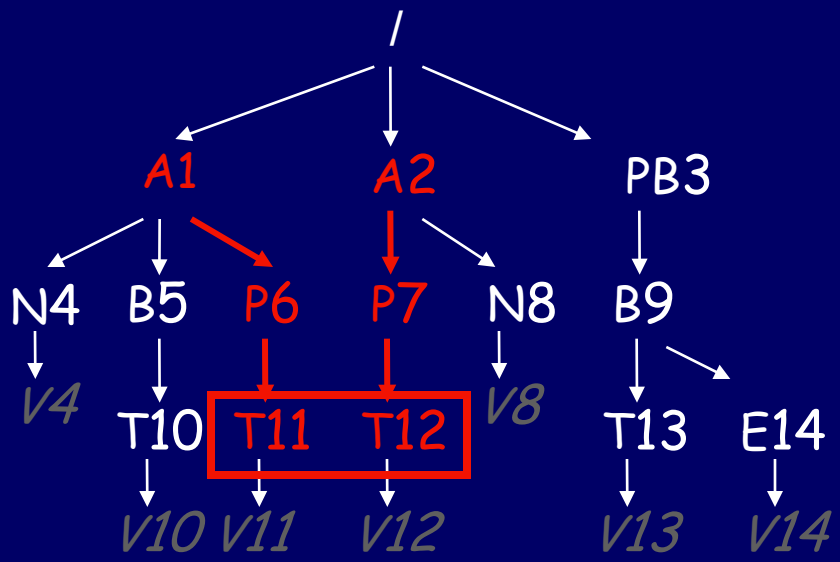
Given an XML document, the *value* of a path expression *p* is a set of elements (= XML subtrees)

Path Expressions



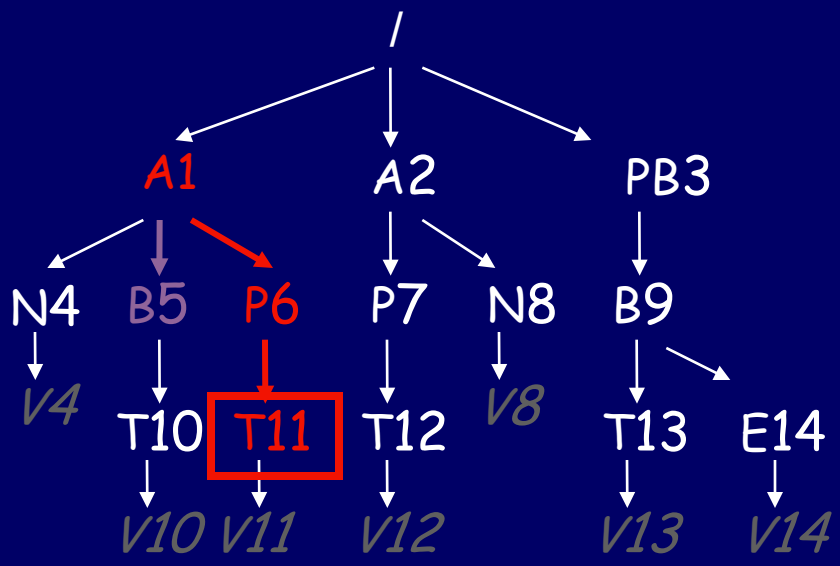
- XPath expressions
  - Simple: /A/P/T
  - Branching: /A[B]/P/T
  - Values: /A/P/T[=v11]
- Result is a <u>set</u>

Path Expressions



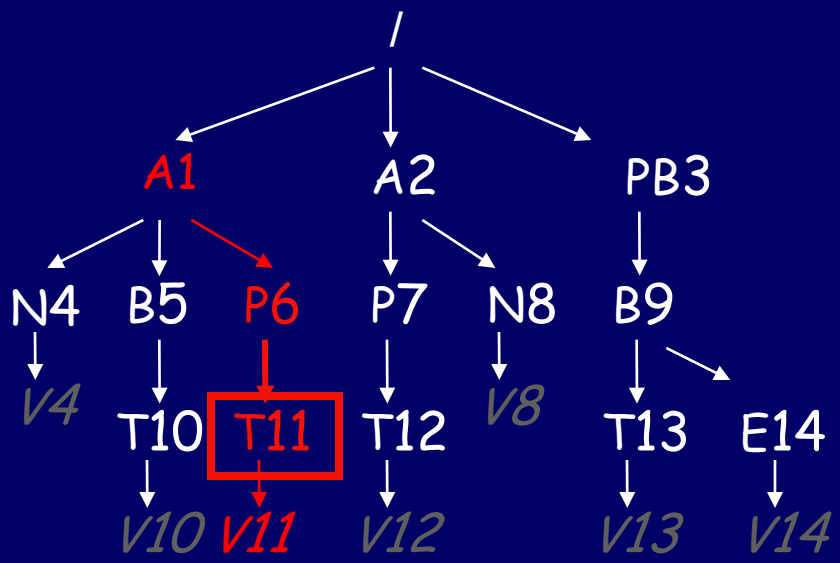
- XPath expressions
  - Simple: /A/P/T
  - Branching: /A[B]/P/T
  - Values: /A/P/T[=v11]
- Result is a set

Path Expressions



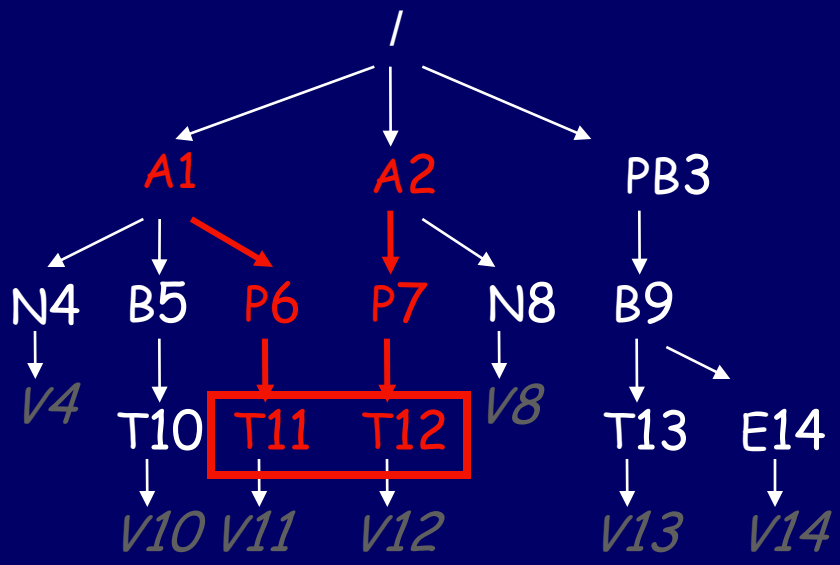
- XPath expressions
  - Simple: /A/P/T
  - Branching: /A[B]/P/T
  - Values: /A/P/T[=v11]
- Result is a <u>set</u>

Path Expressions



- XPath expressions
  - Simple: /A/P/T
  - Branching: /A[B]/P/T
  - Values: /A/P/T[=v11]
- Result is a set

Path Expressions



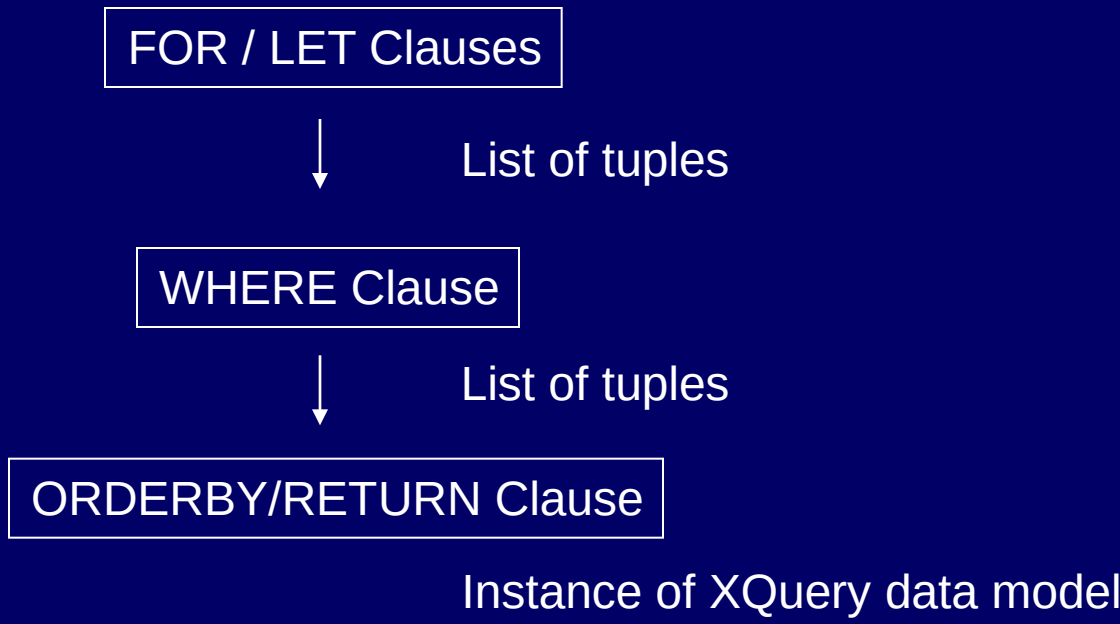
- XPath expressions
  - Simple: /A/P/T
  - Branching: /A[B]/P/T
  - Values: /A/P/T[=v11]
- Result is a set

XPath Syntax

- Path wildcards
  - // = descendant at any level (or self)
  - * = any (single) tag
  - Example: **/booklist//lastname**
- Query attributes and attribute content
  - Use "@"
  - Examples: **/booklist//book[@format="Paperback"], /booklist//book/@genre**
- Branching predicates: A[*pred*]
  - Predicate on A's subtree using *logical connectives* (and, or, etc.), *path expressions*, *built-in functions* (e.g., contains()), etc.
  - Example: **//author[contains(./lastname, "Fey")]**

XQuery FLWOR Expressions

- FOR-LET-WHERE-ORDERBY-RETURN = FLWOR



XQUERY - examples

```
<bookstore>
        <book category="COOKING">
                <title lang="en">Everyday Italian</title>
                <author>Giada De Laurentiis</author>
                <year>2005</year>
                <price>30.00</price>
        </book>
        <book category="CHILDREN">
                <title lang="en">Harry Potter</title>
                <author>J K. Rowling</author>
                <year>2005</year>
                <price>29.99</price>
        </book>
        <book category="WEB">
                <title lang="en">XQuery Kick Start</title>
                <author>James McGovern</author>
                <author>Per Bothner</author>
                <author>Kurt Cagle</author>
                <author>James Linn</author>
                <author>Vaidyanathan Nagarajan</author>
                <year>2003</year>
                <price>49.99</price>
        </book>
        <book category="WEB">
                <title lang="en">Learning XML</title>
                <author>Erik T. Ray</author>
                <year>2003</year>
                <price>39.95</price>
        </book>
</bookstore>
```

- **Query:**
  doc("books.xml")/bookstore/book/title
- **Result:**

```
<title lang="en">Everyday Italian</title>
<title lang="en">Harry Potter</title>
<title lang="en">XQuery Kick Start</title>
<title lang="en">Learning XML</title>
```

XQUERY - examples

```
<bookstore>
        <book category="COOKING">
                <title lang="en">Everyday Italian</title>
                <author>Giada De Laurentiis</author>
                <year>2005</year>
                <price>30.00</price>
        </book>
        <book category="CHILDREN">
                <title lang="en">Harry Potter</title>
                <author>J K. Rowling</author>
                <year>2005</year>
                <price>29.99</price>
        </book>
        <book category="WEB">
                <title lang="en">XQuery Kick Start</title>
                <author>James McGovern</author>
                <author>Per Bothner</author>
                <author>Kurt Cagle</author>
                <author>James Linn</author>
                <author>Vaidyanathan Nagarajan</author>
                <year>2003</year>
                <price>49.99</price>
        </book>
        <book category="WEB">
                <title lang="en">Learning XML</title>
                <author>Erik T. Ray</author>
                <year>2003</year>
                <price>39.95</price>
        </book>
</bookstore>
```

- **Query:**

```
doc("books.xml")/bookstore/
                book[price<30]
```

- **Result:**

```
<book category="CHILDREN">
        <title lang="en">Harry Potter</title>
        <author>J K. Rowling</author>
        <year>2005</year>
        <price>29.99</price>
</book>
```

XQUERY - examples

```
<bookstore>
        <book category="COOKING">
                <title lang="en">Everyday Italian</title>
                <author>Giada De Laurentiis</author>
                <year>2005</year>
                <price>30.00</price>
        </book>
        <book category="CHILDREN">
                <title lang="en">Harry Potter</title>
                <author>J K. Rowling</author>
                <year>2005</year>
                <price>29.99</price>
        </book>
        <book category="WEB">
                <title lang="en">XQuery Kick Start</title>
                <author>James McGovern</author>
                <author>Per Bothner</author>
                <author>Kurt Cagle</author>
                <author>James Linn</author>
                <author>Vaidyanathan Nagarajan</author>
                <year>2003</year>
                <price>49.99</price>
        </book>
        <book category="WEB">
                <title lang="en">Learning XML</title>
                <author>Erik T. Ray</author>
                <year>2003</year>
                <price>39.95</price>
        </book>
</bookstore>
```

- **Query:**

```
for $x in doc("books.xml")/bookstore
    /book where $x/price>30 order
    by $x/title return $x/title
```

- **Result:**

```
<title lang="en">Learning XML</title>
<title lang="en">XQuery Kick Start</title>
```

FOR vs. LET

- FOR $x IN path-expression
  - Binds $x in turn to each element in the expression
- LET $x := path-expression
  - Binds $x to the *entire list of elements* in the expression
  - Useful for common sub-expressions and for aggregations

FOR vs. LET: Example

```
FOR $x IN document("bib.xml")/bib/book
        RETURN <result> $x </result>
```

Returns:

<result> <book>...</book></result>
<result> <book>...</book></result>
<result> <book>...</book></result>
...

Notice that result has several elements

```
LET $x := document("bib.xml")/bib/book
        RETURN <result> $x </result>
```

Returns:

<result> <book>...</book>
<book>...</book>
<book>...</book>
...
</result>

Notice that result has exactly one element

XQuery Example 1

Find all book titles published after 1995:

```
FOR $x IN document("bib.xml")/bib/book
    WHERE $x/year > 1995
    RETURN $x/title
```

Result:

<title> abc </title>
<title> def </title>
<title> ghi </title>

XQuery Example 2

For each author of a book by Morgan Kaufmann, list all books she published:

```
FOR $a IN distinct(
 document("bib.xml"/bib/book[publisher="Morgan Kaufmann"]/author))
RETURN <result>
                $a,
                FOR $t IN /bib/book[author=$a]/title
                RETURN $t
            </result>
```

distinct = a function that eliminates duplicates (after converting inputs to atomic values)

Results for Example 2

```
<result>
      <author>Jones</author>
      <title> abc </title>
      <title> def </title>
</result>
<result>
       <author> Smith </author>
       <title> ghi </title>
</result>
```

Observe how nested structure of result elements is determined by the nested structure of the query.