Outline of Talk

- **XML Databases**
  - Discuss XML in the context of Databases
  - Relational Databases vs. XML Databases

- **XML Database Languages**
  - XPath
  - XUpdate
A database is a collection of data that is organized so that its contents can easily be accessed, managed, and updated.

The most prevalent type of database is the relational database, a tabular database in which data is defined so that it can be reorganized and accessed in a number of different ways:

- Microsoft Access
- Oracle
- Sybase
- IBM’s DB2
“Relational databases organize data in a tabular, or grid-like fashion, and use relational linking in order to expose hierarchical constraints on the data”[8]

XML documents are organized as hierarchical tree structures
“Unfortunately, while it's generally very easy to map relational data into XML, trying to map XML, which can be very complex and freeform, into relational data can be incredibly difficult and lossy” [8]

This means ugly routines for mapping XML data structures into RDB structures

– If you must do this, there is documentation available to you at http://www.xml.com/pub/a/2001/06/20/databases.html
– Fortunately, there is a simpler alternative for most of us…
Xindice is a native XML-Database

“The benefit of a native solution is that you don't have to worry about mapping your XML to some other data structure. You just insert the data as XML and retrieve it as XML…[which is] especially valuable when you have very complex XML structures that would be difficult or impossible to map to a more structured database.” [2]
Xindice stores and indexes compressed XML documents

“It was not designed to store and manage single monster sized documents, where one document is treated as a set of mini documents. It was specifically designed for managing many small to medium sized documents.” [8]
Because native XML databases store data in a different way from relational databases, there are different languages used to query and update the database.

- **Query**
  - To retrieve a specified portion of a database

- **Update**
  - To add or modify a portion of a database
For relational databases, the Structured Query Language (SQL) is often used to perform both queries and updates.

For Xindices (and other XML native databases):
- queries are specified using the XPath language
- updates are defined using the XUpdate language

Let’s take a look at both XPath and XUpdate in more detail.
XPath
XPath is a (W3C-standard) language used to address a section of an XML document

“XPath uses a compact, non-XML syntax to facilitate use of XPath within URIs”[6]

A single string of XPath can address any section of an XML document

The next few slides show the basics of XPath by giving some basic examples
Some example XPath

- If the XPath string starts with // then all elements in the document which fulfill the criteria which follow the // are selected.
- * selects all elements located by the preceding path
- //*
  - Select all elements
//DDD/BBB
- Select all elements BBB which are direct children of DDD

If the XPath was /DDD/BBB, no nodes would be selected because there is no root DDD element which contains a BBB element

```xml
<AAA>
  <BBB/>
  <CCC/>
  <DDD>
    <BBB/>
  </DDD>
</DDD>
<CCC>
  <DDD>
    <BBB/>
    <BBB/>
  </DDD>
</CCC>
</AAA>
```
Example XPath[5]

- Select all elements BBB which have exactly 3 ancestors
Some example XPath[5]

- /AAA/BBB[1]
  - Select the first BBB child of element AAA

<AAA>
  <BBB/>
  <BBB/>
  <BBB/>
  <BBB/>
</AAA>
An example XPath Function[5]

/AAA/BBB[last()]
- Select the last BBB child of element AAA
- This is an example of an XPath “function”
An example XPath Function[5]

- //*[count(BBB)=2]
  - Select elements which have two children BBB

```xml
<AAA>
  <CCC>
    <BBB/>
    <BBB/>
    <BBB/>
  </CCC>
  <DDD>
    <BBB/>
    <BBB/>
    <BBB/>
  </DDD>
  <EEE>
    <DDD/>
  </EEE>
</AAA>
```
Combining XPath

- | is used to combine XPath queries
- //CCC | //BBB
  - Select all elements CCC and all elements BBB
Some example XPath

- `/AAA/BBB/descendant::*`
  - Select all descendants of `/AAA/BBB`
//DDD/parent::*
  - Select all parents of DDD element

Some example XPath\[^5\]

- `<AAA>`
  - `<BBB>`
    - `<DDD>`
      - `<CCC>`
        - `<DDD/>`
        - `<EEE/>`
        - `</CCC>`
    - `</DDD>`
  - `</BBB>`
  - `<CCC>`
    - `<DDD>`
      - `<EEE>`
        - `<DDD>`
        - `<FFF/>`…
Some example XPath[5]

- /AAA/BBB/DDD/CCC
- /EEE/ancestor::*
  - Select all elements given in this absolute path
The div operator performs floating-point division.

The mod operator returns the remainder from a truncating division.

The floor function returns the largest integer that is not greater than the argument.

The ceiling function returns the smallest integer that is not less than the argument.
//BBB[position() mod 2 = 0 ]

- Selects every second BBB node

```xml
<AAA>
  <BBB/>
  <BBB/>
  <BBB/>
  <BBB/>
  <BBB/>
  <BBB/>
  <BBB/>
  <BBB/>
  <BBB/>
  <BBB/>
  <CCC/>
  <CCC/>
  <CCC/>
</AAA>
```
Xindice’s API directly supports XPath queries
Performing such a query involves a few simple lines of java

```java
String xpath = new String();
xpath = "//person/name/*"; //the XPath string

XPathQueryService xp_service =
    col.getService("XPathQueryService", "1.0");
ResourceSet resultSet = xp_service.query(xpath);
```
XUpdate is an XML-syntax that is used to specify modifications of XML documents.

It is part of the XML:DB Project (http://www.xmldb.org/xupdate/).

XUpdate uses XPath expressions to select nodes for processing.

Let’s look at an example…
This example XUpdate adds a middle name element before the last name element

```xml
<xupdate:modifications version="1.0"
    xmlns:xupdate="http://www.xmldb.org/xupdate">
    <xupdate:insert-before
        select="/addresses/address[@id = 1]/name/last" >
        <xupdate:element
            name="middle">Lennox</xupdate:element>
    </xupdate:insert-before>
</xupdate:modifications>
```
All XUpdate commands have a select attribute which uses XPath to specify an XML node to act upon

```xml
<xupdate:modifications version="1.0"
    xmlns:xupdate="http://www.xmldb.org/xupdate">
    <xupdate:insert-before
        select="/addresses/address[@id = 1]/name/last" >
        <xupdate:element
            name="middle">Lennox</xupdate:element>
    </xupdate:insert-before>
</xupdate:modifications>
```
<xupdate:modifications version="1.0"
    xmlns:xupdate="http://www.xmldb.org/xupdate">

    <xupdate:append select="/addresses">
        <xupdate:element name="address">
            <new_xml_block>
                <valid_new_xml_goes_here/>
            </new_xml_block>
        </xupdate:element>
    </xupdate:append>

</xupdate:modifications>
A few things you can do with XUpdate

- Insert Element/Attribute/Comment
- Update Element/Attribute/Comment
- Delete Element/Attribute/Comment
- Rename Element/Attribute
- Update/Delete Text Content of an Element
- Insert XML Block
- Append Element
- Copy a Node
- Move a Node
Xindice’s API directly supports XUpdate updates
Performing such an update involves a few simple lines of java (which are very similar to those for XPath queries)

String update=new String();
update = “XUPDATE STRING”;

XUpdateQueryService xupdate_service =
    data.getService("XUpdateQueryService","1.0");
xupdate_service.update(update);
XPath and XUpdate are the XML-Database languages which are supported by Xindice.

Performing XPath queries or XUpdate updates using the Xindice API is easy.

XQuery is a developing XML alternative to XPath (which has the non XML, path-like structure):
  - More SQL like database queries
  - Not yet supported by Xindice
[2] Xindice homepage  
http://xml.apache.org/xindice/

http://www.zvon.org/xxl/XPathTutorial/General/examples.html

http://www.w3.org/TR/xpath
[7] XUpdate Example Use Cases Page
http://www.xml databases.org/projects/XUpdate-UseCases/

[8] Xindice FAQ
http://xml.apache.org/xindice/FAQ