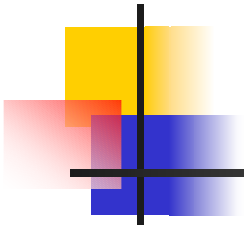
Xquery Tutorial

Craig Knoblock
University of Southern California

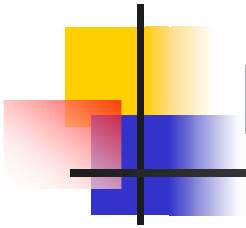


References

- XQuery 1.0: An XML Query Language
  - www.w3.org/TR/xquery/
- XML Query Use Cases
  - www.w3.org/TR/xmlquery-use-cases
- Demonstration site:
  - http://www.cocoonhive.org/xquery/xqueryform.html
- Xquery Tutorial by Fankhauser & Wadler
  - www.research.avayalabs.com/user/wadler/papers/xquery-tutorial/ xquery-tutorial.pdf

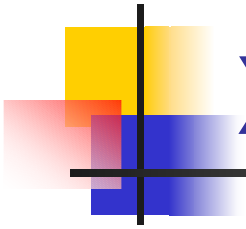
Example XML Document: bib.xml

```
<bib>
    <book year="1994">
        <title>TCP/IP Illustrated</title>
        <author><last>Stevens</last><first>W.</first></author>
        <publisher>Addison-Wesley</publisher>
        <price> 65.95</price>
    </book>
    <book year="1992">
        <title>Advanced Programming in the Unix
          environment</title>
          <author><last>Stevens</last><first>W.</first></author>
        <publisher>Addison-Wesley</publisher>
        <price>65.95</price>
    </book>
```



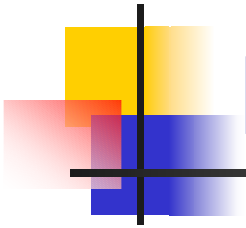
Example XML Document: bib.xml

```
<book year="2000">
      <title>Data on the Web</title>
      <author><last>Abiteboul</last><first>Serge</first></author>
      <author><last>Buneman</last><first>Peter</first></author>
      <author><last>Suciu</last><first>Dan</first></author>
      <publisher>Morgan Kaufmann Publishers</publisher>
      <price> 39.95</price>
</book>
<book year="1999">
      <title>The Economics of Technology and Content for Digital TV</title>
      <editor> <last>Gerbarg</last><first>Darcy</first> <affiliation>CITI</affiliation>
        </editor>
      <publisher>Kluwer Academic Publishers</publisher>
      <price>129.95</price>
</book>
</bib>
```



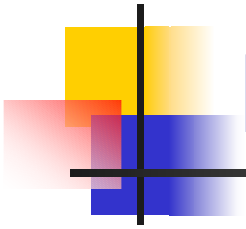
Xquery Overview

- Xquery is an expression language
  - Every statement evaluates to some result
    - Let $x := 5 let $y := 6 return 10*$x+$y
    - Evaluates to 56
- Primitive types
  - Number, boolean, strings, dates, times, durations, and XML types
- Derived types
  - Restrictions of other types (e.g., range types)



Nodes and Expressions

- Various functions create or return nodes
  - Document function reads an XML file
    - document("http://www.isi.edu/info-agents/courses/iiweb/bib.xml")/bib
    - We will use /bib throughout, but you must use the expansion to run the demo
  - Element constructor creates a node:
    - <doc><par>Blah Blah</par></doc>
  - Use curly braces to embed Xquery expressions inside an element constructor



Path Expressions

- Xquery uses path expressions from Xpath (a W3C standard)
- Let $book :=
  document("mybook.xml)/book
  return $book/chapter
- /book selects the child elements named book
- /book/chapter selects the child elements of the top-level book elements

FLWR Expressions

- For/Let, Where, Result Expressions
- <html> {
  let $book := document("mybook.xml")/book
  for $ch in $book/chapter
  where $book/chapter/num < 10
  return <h2>{$ch/title}</h2>
  </html>

Projection

- Return the names of all authors of books

  /bib/book/author

  =

<author><last>Stevens</last><first>W.</first></author>
<author><last>Stevens</last><first>W.</first></author>
<author><last>Abiteboul</last><first>Serge</first></author>
<author><last>Buneman</last><first>Peter</first></author>
<author><last>Suciu</last><first>Dan</first></author>

Project (cont.)

- The same query can also be written as a for loop

/bib/book/author

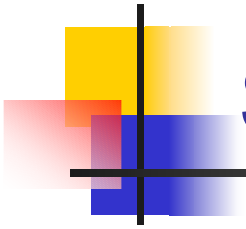
=

for $bk in /bib/book return

for $aut in $bk/author return $aut

=

<author><last>Stevens</last><first>W.</first></author>
<author><last>Stevens</last><first>W.</first></author>
<author><last>Abiteboul</last><first>Serge</first></author>
<author><last>Buneman</last><first>Peter</first></author>
<author><last>Suciu</last><first>Dan</first></author>



Selection

- Return the titles of all books published before 1997

/bib/book[@year < "1997"]/title

=

<title>TCP/IP Illustrated</title>

<title>Advanced Programming in the Unix environment</title>

Selection (cont.)

- Return the titles of all books published before 1997

```
/bib/book[@year < "1997"]/title
=
for $bk in /bib/book
   where $bk/@year < "1997"
      return $bk/title
=
<title>TCP/IP Illustrated</title>
<title>Advanced Programming in the Unix
   environment</title>
```

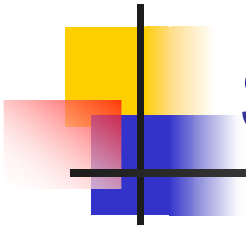
Selection (cont.)

- Return book with the title “Data on the Web”

/bib/book[title = "Data on the Web"]

=

```
<book year="2000">
    <title>Data on the Web</title>
    <author><last>Abiteboul</last><first>Serge</first></author
      >
    <author><last>Buneman</last><first>Peter</first></author
      >
    <author><last>Suciu</last><first>Dan</first></author>
    <publisher>Morgan Kaufmann Publishers</publisher>
    <price> 39.95</price>
</book>
```



Selection (cont.)

- Return the price of the book “Data on the Web”

/bib/book[title = "Data on the Web"]/price

=

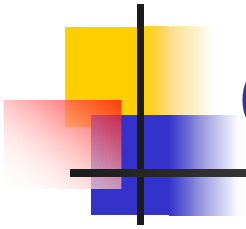
<price> 39.95</price>

How would you return the book with a price of $39.95?

Selection (cont.)

- Return the book with a price of $39.95

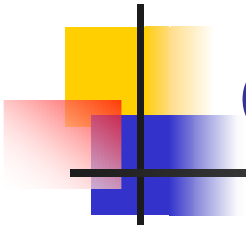
```
for $bk in /bib/book
   where $bk/price = " 39.95"
      return $bk
=
      <book year="2000">
            <title>Data on the Web</title>
            <author><last>Abiteboul</last><first>Serge</first></author>
            <author><last>Buneman</last><first>Peter</first></author>
            <author><last>Suciu</last><first>Dan</first></author>
            <publisher>Morgan Kaufmann Publishers</publisher>
            <price> 39.95</price>
      </book>
```



Construction

- Return year and title of all books published before 1997

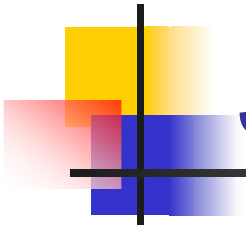
```
for $bk in /bib/book
   where $bk/@year < "1997"
   return <book>{ $bk/@year, $bk/title }</book>
=
<book year="1994">
  <title>TCP/IP Illustrated</title>
</book>
<book year="1992">
  <title>Advanced Programming in the Unix environment</title>
</book>
```



Grouping

- Return titles for each author

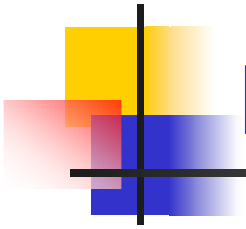
```
for $author in distinct(/bib/book/author/last) return
<author name={ $author/text() }>
   { /bib/book[author/last = $author]/title }
</author>
=
<author name="Stevens">
  <title>TCP/IP Illustrated</title>
  <title>Advanced Programming in the Unix environment</title>
</author>
<author name="Abiteboul">
  <title>Data on the Web</title>
</author>
…
```



Join

- Return the books that cost more at amazon than fatbrain

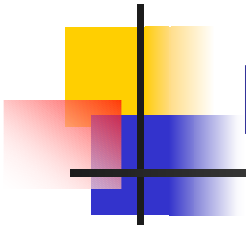
```
Let $amazon := document(http://www.amazon.com/books.xml),
Let $fatbrain := document(http://www.fatbrain.com/books.xml)
For $am in $amazon/books/book,
      $fat in $fatbrain/books/book
Where $am/isbn = $fat/isbn
    and $am/price > $fat/price
Return <book>{ $am/title, $am/price, $fat/price }<book>
```



Example Query 1

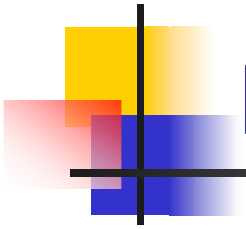
```
<bib>
 { for $b in /bib/book
   where $b/publisher = "Addison-Wesley" and $b/@year > 1991
   return <book year={ $b/@year }>
                    { $b/title }
          </book> }
</bib>
```

What does this do?



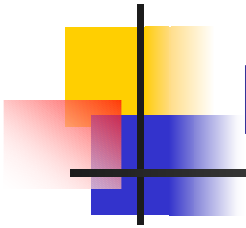
Result Query 1

```
<bib>
  <book year="1994">
    <title>TCP/IP Illustrated</title>
  </book>
  <book year="1992">
    <title>Advanced Programming in the Unix environment</title>
  </book>
</bib>
```



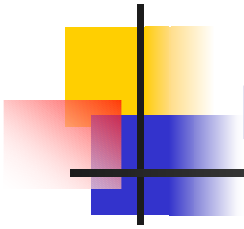
Example Query 2

```
<results>
{ for $b in document("http://www.bn.com/bib.xml")/bib/book,
        $t in $b/title,
        $a in $b/author
    return
        <result>
            { $t }
            { $a }
        </result> }
</results>
```



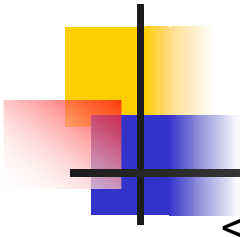
Result Query 2

```
<results>
  <result><title>TCP/IP Illustrated</title>
          <last>Stevens </last>
  </result>
  <result><title>Advanced Programming in the Unix environment</title>
          <last>Stevens</last>
  </result>
  <result><title>Data on the Web</title>
          <last>Abiteboul</last>
  </result>
  <result> <title>Data on the Web</title>
           <last>Buneman</last>
  </result>
  <result><title>Data on the Web</title>
          <last>Suciu</last>
  </result>
</results>
```



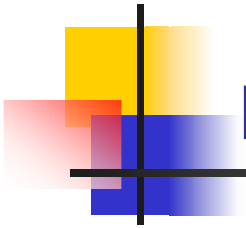
Example Query 3

```
<books-with-prices>
{
    for $b in document("http://www.bn.com/bib.xml")//book,
        $a in document("http://www.amazon.com/reviews.xml")//entry
    where $b/title = $a/title
    return
        <book-with-prices>
            { $b/title }
                <price-amazon>{ $a/price/text() }</price-amazon>
                <price-bn>{ $b/price/text() }</price-bn>
        </book-with-prices>
}
</books-with-prices>
```



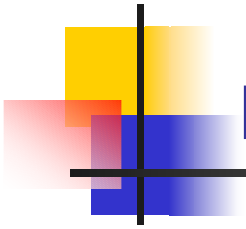
Result Query 3

```
<books-with-prices>
  <book-with-prices>
    <title>TCP/IP Illustrated</title>
    <price-amazon>65.95</price-amazon>
    <price-bn> 65.95</price-bn>
  </book-with-prices>
  <book-with-prices>
    <title>Advanced Programming in the Unix environment</title>
    <price-amazon>65.95</price-amazon>
    <price-bn>65.95</price-bn>
  </book-with-prices>
  <book-with-prices>
    <title>Data on the Web </title>
    <price-amazon>34.95</price-amazon>
    <price-bn> 39.95</price-bn>
  </book-with-prices>
</books-with-prices>
```



Example Query 4

```
<bib>
 { for $b in document("www.bn.com/bib.xml")//book
   where $b/publisher = "Addison-Wesley" and $b/@year > "1991"
   return <book> { $b/@year } { $b/title } </book>
   sortby (title) }
</bib>
```



Example Result 4

```
<bib>
  <book year="1992">
  <title>Advanced Programming in the Unix environment</title>
  </book>
  <book year="1994">
  <title>TCP/IP Illustrated</title>
  </book>
</bib>
```