

Xquery Tutorial

Dan Goldberg & 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
- What is XQuery: Katz, 2004. "XQuery from the Experts"
 - <http://www.gnu.org/software/qexo/XQuery-Intro.html>
- XQuery Tutorial
 - http://www.datadirect.com/developer/xquery/xquery_tutorial/index.ssp
- XQuery Tutorial
 - <http://www.w3schools.com/xquery/default.asp>
- XQuery Tutorial
 - http://www.stylusstudio.com/tutorials/xquery_tutorials.html

Availability

- Xquery Interpreter:
 - Qexo
(<http://www.gnu.org/software/qexo/>)
- Advice:
 - Case matters
 - Write your queries incrementally!
 - It is hard to debug entire queries

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>
<collection year="2007">
  <title>Handbook of GIS</title>
  <editor><last>Wilson</last><first>John</first></editor>
  <publisher>Blackwell</publisher>
  <price>124.95</price>
</collection>
<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>
</bib>
```

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" pages="213">
  <title>The Economics of Technology and Content for Digital TV</title>
  <editor><last>Gerburg</last><first>Darcy</first> <affiliation>CITI</affiliation> </editor>
  <publisher>Kluwer Academic Publishers</publisher>
  <price>129.95</price>
</book>
<collection year="2007">
  <title>Proceedings of AAG</title>
  <editor><last>Goodchild</last><first>M</first><middle>F</middle></editor>
  <publisher>Morgan Kaufmann Publishers</publisher>
  <price>83.99</price>
</collection>
</bib>
```

Xquery Overview

- Xquery is an expression language
 - Every statement evaluates to some result
 - (ex1.xql)
 - let \$x := 5 let \$y := 6 return 10*\$x+\$y
 - Evaluates to 56
- Primitive types
 - Number, boolean, strings, dates, times, durations, and XML types

Nodes and Expressions

- Various functions create or return nodes
 - Document function reads an XML file
 - `doc("bib.xml")`
 - We will use `doc("bib.xml")` 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)
- (Ex2.xql – note the lowercase "let")
- `let $b := doc("bib.xml")`
`return <result>{$b/bib/book}</result>`
- `/book` selects the child elements named book
- `/book/author` selects the author elements of the top-level book elements

Path Expressions (cont.)

- (Ex3.xql)
 - `//book`
 - returns all book elements that appear anywhere in the document - including child elements
- (Ex4.xql)
 - `//book[author/last = "Stevens"]`
 - all book elements with author = "Hunter"
 - ex4.xql:2:40: invalid character '\u201c' [XPST0003] – Oh No!!!!
 - **Make sure you use plain ascii quotes!!**
- (Ex5.xql)
 - `//book[@pages]`
 - all book elements that have a pages attribute
- (Ex6.xql)
 - `//book[@year > 1999]`
 - book elements with attribute year > 1999

Advanced Path Expressions

- (Ex7.xql)
 - `//book/(author | editor)` – returns all author or editor elements from current node
- (Ex8.xql)
 - `(//book | //collection)[publisher = $pub]` – books for collections where publisher equals \$pub variable

```
let $d := doc("bib.xml")
let $pub := "Morgan Kaufmann Publishers"
return <result>{(
  $d//book | $d//collection)[publisher = $pub
}</result>
```
- (Ex9.xql)
 - `//book[1]/title/text()` – returns the text nodes of the first book element

FLWOR Expressions

- For/Let, Where, Order by, Result Expressions (Ex10.xql)

```
<html>{
let $d := doc("bib.xml")
for $b in $d/bib/book
where $b[@year > 1998]
order by $b/publisher
return <book>{$b/title, $b/price,
  $b/publisher}</book>
}</html>
```

Projection

- Return the names of all authors of books
- Ex11.xql

```
let $d := doc("bib.xml")
return <result>{$d/bib/book/author}</result>
=
<result>
<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>
</result>
```

Project (cont.)

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

```
$d/bib/book/author
```

```
=
```

```
(Ex12.xql)
```

```
for $bk in doc("bib.xml")/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

- (ex13.xql)

```
$d/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
```

```
=
```

```
(Ex14.xql)
```

```
for $bk in doc("bib.xml")/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"

- (Ex15.xql)

```
$d/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"

- (Ex16.xql)

```
$d/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

```
(Ex17.xql)
```

```
for $bk in doc("bib.xml")/bib/book
```

```
where $bk/price = "39.95"
```

```
return $bk
```

```
=
```

```
(Ex18.xql)
```

```
let $d:= doc("bib.xml")
```

```
return $d/bib/book[price = "39.95"]
```

```
=
```

```
<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 (Ex19a.xql)

```
for $bk in doc("bib.xml")/bib/book
where $bk/@year < "1997"
return <book> $bk/@year, $bk/title </book>
=
<book>
$bk/@year, $bk/title
</book>
<book year="1992">
$bk/@year, $bk/title
</book>
```

Oh No !!!! - Don't forget the brackets !!!!

Construction

- Return year and title of all books published before 1997 (Ex19b.xql)

```
for $bk in doc("bib.xml")/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 (Ex20.xql)

```
<result>{
let $d:= doc("bib.xml")
for $l in distinct-values($d//author/last)
return
<author name="{ $l }">
{ $d/bib/book[author/last = $l]/title }
</author>
}</result>
```

Grouping Cont.

```
=
<result>
<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>
</result>
```

Join

Return the books that cost less at bib than bib2 (Ex21.xql)

```
for
$book1 in doc("bib.xml")//book,
$book2 in doc("bib2.xml")//book[title=$book1/title and
price<$book1/price]
return
<book>
{ $book1/title, $book1/price, $book2/price }
</book>
```

Join

Return the books that cost less at bib than bib2 (Ex22.xql)

```
let
$bib1 := doc("bib.xml"),
$bib2 := doc("bib2.xml")
for
$book1 in $bib1/bib/book,
$book2 in $bib2/bib/book
where
$book1/title = $book2/title
and $book1/price < $book2/price
return <book>{ $book1/title, $book1/price, $book2/price }</book>
```

Functions

```
(Ex23.xql)
declare function local:Total($bib) as xs:double {
  sum($bib/book/price/text());
};

let $d := doc("bib.xml")
return
<total> {local:Total($d/bib)}</total>
```

Example Query 1

```
(Ex24.xql)
<bib>
{
  for $b in doc("bib.xml")/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

```
(ex25.xql)
<results>
{ for $b in doc("bib.xml")/bib/book,
  $t in $b/title,
  $a in $b/author/last
  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

```
(ex26.xql)
<bib>
{ for $b in doc("bib.xml")/bib/book
  where $b/publisher = "Addison-Wesley" and $b/@year > "1991"
  order by $b/title
  return <book> { $b/@year } { $b/title } </book>
}
</bib>
```

Example Result 3

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

Example 4

(ex27.xql)

```
for $b in doc("bib.xml")//book[author and count(author) = 1 and author/last and
author/first]
order by $b/author/last
return
  concat(
    $b/author/last/text(),
    ", ",
    substring($b/author/first/text(), 1, 1),
    ". ",
    $b/@year,
    ". ",
    $b/title/text(),
    ". ",
    $b/publisher,
    "\n "
  )
```

Example 4 results

- Stevens, W. 1994. TCP/IP Illustrated. Addison-Wesley\n Stevens, W. 1992. Advanced Programming in the Unix environment. Addison-Wesley\n

Example 5

(ex28.xql)

```
for $b in doc("bib.xml")//book[author and count(author) = 1 and author/last and
author/first]
order by $b/author/last
return
  <bibEntry>
  {
    concat(
      $b/author/last/text(),
      ", ",
      substring($b/author/first/text(), 1, 1),
      ". ",
      $b/@year,
      ". ",
      $b/title/text(),
      ". ",
      $b/publisher
    )
  }
</bibEntry>
```

Example 5 results

```
<bibEntry>Stevens, W. 1994. TCP/IP
  Illustrated. Addison-Wesley
</bibEntry>
<bibEntry>Stevens, W. 1992. Advanced
  Programming in the Unix environment.
  Addison-Wesley</bibEntry>
```