Example Queries with XQuery

Sample XML Data

```xml
<BOOKS>
  <BOOK YEAR="1999">
    <AUTHOR>Abiteboul</AUTHOR>
    <AUTHOR>Buneman</AUTHOR>
    <AUTHOR>Suciu</AUTHOR>
    <TITLE>Data on the Web</TITLE>
    <REVIEW>A <EM>fine</EM> book.</REVIEW>
    <PRICE>40.00</PRICE>
    <SHIPPING>10.00</SHIPPING>
  </BOOK>
  <BOOK YEAR="2002">
    <AUTHOR>Buneman</AUTHOR>
    <TITLE>XML in Scotland</TITLE>
    <REVIEW>The <EM>best</EM> ever!</REVIEW>
    <PRICE>45.00</PRICE>
  </BOOK>
</BOOKS>
```
Projection

• Path expression (Example 0a):

  (: Projection with Path expression: “Return all authors of all books.” :)  
doc("bib.xml")/BOOKS/BOOK/AUTHOR

• FLWOR expression (Example 0b):

  (: Projection with FLWOR expression: "Return all authors of all books." :)  
  let $bib := doc("bib.xml")
  for $dot1 in $bib/BOOKS return
  for $dot2 in $dot1/BOOK return
  $dot2/AUTHOR

Projection

• Result:

  <AUTHOR>Abiteboul</AUTHOR>
  <AUTHOR>Buneman</AUTHOR>
  <AUTHOR>Suciu</AUTHOR>
  <AUTHOR>Buneman</AUTHOR>
Selection (Attribute)

- Path expression (Example 1a):

  (: Selection with Path expression: "Return titles of all books published before 2000." :) 
  doc("bib.xml")/BOOKS/BOOK[@YEAR < 2000]/TITLE

- FLWOR expression (Example 1b):

  (: Selection with FLWOR expression. "Return titles of all books published before 2000." :) 
  let $bib := doc("bib.xml") 
  for $book in $bib/BOOKS/BOOK 
  where $book/@YEAR < 2000 
  return $book/TITLE

Selection

- Result:

  <TITLE>Data on the Web</TITLE>
Selection (Element)

- Path expression (Example 2a):

  (: Selection with Path expression: "Return book with title 'Data on the Web'." :) 
doc("bib.xml")/BOOKS/BOOK[TITLE = "Data on the Web"]

- FLWOR expression (Example 2b):

  (: Selection with FLWOR expression: "Return book with title 'Data on the Web'." :) 
  let $bib := doc("bib.xml")
  for $book in $bib/BOOKS/BOOK 
  where $book/TITLE = "Data on the Web"
  return <BOOKS>{ $book }</BOOKS>

Selection

- Result:

  <BOOKS>
  <BOOK YEAR="1999">
  <AUTHOR>Abiteboul</AUTHOR>
  <AUTHOR>Buneman</AUTHOR>
  <AUTHOR>Suciu</AUTHOR>
  <TITLE>Data on the Web</TITLE>
  <REVIEW>A <EM>fine</EM> book.</REVIEW>
  <PRICE>40.00</PRICE>
  <SHIPPING>10.00</SHIPPING>
  </BOOK>
  </BOOKS>
Construction

• Example 3a:

(: Construction. "Return year and title of all books published before 2000." :) 
let $bib := doc("bib.xml")
for $book in $bib/BOOKS/BOOK
where $book/@YEAR < 2000
return <BOOK>{ $book/@YEAR, $book/TITLE }</BOOK>

• Result of 3a:

<BOOK YEAR="1999">
<TITLE>Data on the Web</TITLE>
</BOOK>

• Example 3b:

let $bib := doc("bib.xml")
for $book in $bib/BOOKS/BOOK
where $book/@YEAR < 2000
return <BOOK YEAR="{ $book/@YEAR }"> { $book/TITLE } </BOOK>

• Result of 3b:

<BOOK YEAR="1999">
<TITLE>Data on the Web</TITLE>
</BOOK>
Example 4a:

(: Grouping. "Return titles for each author." :) )

<BIB>
let $bib := doc("bib.xml")
for $author in $bib/BOOKS/BOOK/AUTHOR
return
<AUTHOR NAME="{ $author }">
{ $bib/BOOKS/BOOK[AUTHOR = $author]/TITLE }
</AUTHOR>
</BIB>

Result:

<BIB>
<AUTHOR NAME="Abiteboul">
<TITLE>Data on the Web</TITLE>
</AUTHOR>
<AUTHOR NAME="Buneman">
<TITLE>Data on the Web</TITLE>
<TITLE>XML in Scotland</TITLE>
</AUTHOR>
<AUTHOR NAME="Suciu">
<TITLE>Data on the Web</TITLE>
</AUTHOR>
<AUTHOR NAME="Buneman">
<TITLE>Data on the Web</TITLE>
<TITLE>XML in Scotland</TITLE>
</AUTHOR>
</BIB>
Missing Data

• Example 5a:

( : "Books costing $50.00, where missing shipping is unknown." : )
let $bib := doc("bib.xml")
for $book in $bib/BOOKS/BOOK
  where $book/PRICE + $book/SHIPPING = 50.00
  return $book/TITLE

• Result:

<TITLE>Data on the Web</TITLE>

Functions

• Example 6a:

( : Simplify book by dropping optional year. :) 

define function local:simple ($b) {
  <BOOK> { $b/AUTHOR, $b/TITLE } </BOOK>
};

let $bib := doc("bib.xml")
for $book in $bib/BOOKS/BOOK
return local:simple($book)
Functions

• Result:

<BOOK>
<AUTHOR>Abiteboul</AUTHOR>
<AUTHOR>Buneman</AUTHOR>
<AUTHOR>Suciu</AUTHOR>
<TITLE>Data on the Web</TITLE>
</BOOK>

<BOOK>
<AUTHOR>Buneman</AUTHOR>
<TITLE>XML in Scotland</TITLE>
</BOOK>

Join

• Example 7a:

(: Join. "Find the faculty that are also instructors of a class." ;)
<result> {
let $uni1 := doc("university.xml"),
$uni2 := doc("university.xml")
for $faculty in $uni1/university/school/dept/faculty/person,
   $instructor in $uni2/university/school/dept/class/instructor/person
   where $faculty/name = $instructor/name
return $faculty
} </result>
Join

• Result:

  <result>
  <person ssn="100-00-0010">
    <name>R. Zimmermann</name>
  </person>
  </result>