XML Query Languages
XPath & XQuery

CSC343 Tutorial

Based on material from

‘XQuery’, by Priscilla Walmsley,

O’Reilly Media Inc, 2007

In this tutorial

XQuery Examples

Selecting, Joining, Sorting, Grouping
using FLWOR expressions

cdf> galax-run *query.xq*

Simple Examples

Example 1

```
for $i in 1 to 3
return <oneEval>{$i}</oneEval>
```

```
let $i := (1 to 3)
return <oneEval>{$i}</oneEval>
```

Difference?

Example 1

```
for $i in 1 to 3
return <oneEval>{$i}</oneEval>
```

```
<oneEval>1</oneEval>, <oneEval>2</oneEval>, <oneEval>3</oneEval>
```

```
let $i := (1 to 3)
return <oneEval>{$i}</oneEval>
```

```
<oneEval>1 2 3</oneEval>
```

Example 2

```
(: double for-loop :)

for $i in (1, 2)
for $j in ("a", "b")
return <oneEval>i is {$i} and j is {$j}</oneEval>

for $i in (1, 2), $j in ("a", "b")
return <oneEval>i is {$i} and j is {$j}</oneEval>
```

Example 2

<oneEval>i is 1 and j is a</oneEval>,

<oneEval>i is 1 and j is b</oneEval>,

<oneEval>i is 2 and j is a</oneEval>,

<oneEval>i is 2 and j is b</oneEval>

XML File – catalog.xml

```
<catalog>
 <product dept="WMN">
  <number>557</number>
  <name language="en">Fleece Pullover</name>
  <colorChoices>navy black</colorChoices>
 </product>
 <product dept="ACC">
  <number>563</number>
  <name language="en">Floppy Sun Hat</name>
 </product>
 <product dept="ACC">
  <number>443</number>
  <name language="en">Deluxe Travel Bag</name>
 </product>
 <product dept="MEN">
  <number>784</number>
  <name language="en">Cotton Dress Shirt</name>
  <colorChoices>white gray</colorChoices>
  <desc>Our <i>favorite</i> shirt!</desc>
 </product>
</catalog>
```

Example 3

(: select products :)

```
for $prod in doc("catalog.xml")/catalog/product[@dept = 'ACC']
return $prod/name
```

```
<name language="en">Floppy Sun Hat</name>,
<name language="en">Deluxe Travel Bag</name>
```

```
for $prod in doc("catalog.xml")/catalog/product[@dept = 'ACC']
return $prod
```

Example 4

```
(: list of products in html :)

<html>
  <h1>Product Catalog</h1>
  <ul> {
     for $prod in doc("catalog.xml")/catalog/product
     return <li>number: {data($prod/number)},
            name: {data($prod/name)}</li>
  } </ul>
</html>
```

Example 4

```
<html>
  <h1>Product Catalog</h1>
  <ul>
    <li>number: 557, name: Fleece Pullover</li>
    <li>number: 563, name: Floppy Sun Hat</li>
    <li>number: 443, name: Deluxe Travel Bag</li>
    <li>number: 784, name: Cotton Dress Shirt</li>
  </ul>
</html>
```

Example 5

```
(:  count number of entries :)

<html>
  <h1>Product Catalog</h1>
  <p>A <i>huge</i> list of {count(doc("catalog.xml")//product)}
          products.</p>
</html>
```

FLWORs

For – Let – Where – Order by – Return

XML File – catalog.xml

```
<catalog>
 <product dept="WMN">
  <number>557</number>
  <name language="en">Fleece Pullover</name>
  <colorChoices>navy black</colorChoices>
 </product>
 <product dept="ACC">
  <number>563</number>
  <name language="en">Floppy Sun Hat</name>
 </product>
 <product dept="ACC">
  <number>443</number>
  <name language="en">Deluxe Travel Bag</name>
 </product>
 <product dept="MEN">
  <number>784</number>
  <name language="en">Cotton Dress Shirt</name>
  <colorChoices>white gray</colorChoices>
  <desc>Our <i>favorite</i> shirt!</desc>
 </product>
</catalog>
```

Example 6

```
(: using the where clause :)

for $prod in doc("catalog.xml")//product
let $prodDept := $prod/@dept
where $prodDept = "ACC" or $prodDept = "WM”
return $prod/name
```

```
<name language="en">Fleece Pullover</name>,
<name language="en">Floppy Sun Hat</name>,
<name language="en">Deluxe Travel Bag</name>
```

Example 7

```
(: same example, basically :)
(: intermingled for and let clauses :)

let $doc := doc("catalog.xml")
for $prod in $doc//product
let $prodDept := $prod/@dept
let $prodName := $prod/name
where $prodDept = "ACC" or $prodDept = "WMN“
return $prodName
```

Example 8

```
(: what does this do? :)
let $prods := doc("catalog.xml")//product
for $d in distinct-values($prods/@dept),
   $n in distinct-values($prods[@dept = $d]/number)
return <result dept="{$d}" number="{$n}"/>
```

How many results?

What if we just removed the first or second “distinct-values”?

Could we have avoided using function distinct-values?

Example 8

<result dept="WMN" number="557"/>,
<result dept="ACC" number="563"/>,
<result dept="ACC" number="443"/>,
<result dept="MEN" number="784"/>

Example 8

```
for $prod in doc("catalog.xml")//product
let $d := $prod/@dept
let $n := data($prod/number)
return <result dept="{$d}" number="{$n}"/>
```

XML File – catalog.xml

```
<catalog>
 <product dept="WMN">
  <number>557</number>
  <name language="en">Fleece Pullover</name>
  <colorChoices>navy black</colorChoices>
 </product>
 <product dept="ACC">
  <number>563</number>
  <name language="en">Floppy Sun Hat</name>
 </product>
 <product dept="ACC">
  <number>443</number>
  <name language="en">Deluxe Travel Bag</name>
 </product>
 <product dept="MEN">
  <number>784</number>
  <name language="en">Cotton Dress Shirt</name>
  <colorChoices>white gray</colorChoices>
  <desc>Our <i>favorite</i> shirt!</desc>
 </product>
</catalog>
```

XML File – order.xml

```
<order num="00299432" date="2006-09-15" cust="0221A">
  <item dept="WMN" num="557" quantity="1" color="navy"/>
  <item dept="ACC" num="563" quantity="1"/>
  <item dept="ACC" num="443" quantity="2"/>
  <item dept="MEN" num="784" quantity="1" color="white"/>
  <item dept="MEN" num="784" quantity="1" color="gray"/>
  <item dept="WMN" num="557" quantity="1" color="black"/>
</order>
```

Example 9

(: join :)

```
for $item in doc("order.xml")//item,
        $product in doc("catalog.xml")//product
where $item/@num = $product/number
return <item num="{$item/@num}"
        name="{$product/name}"
        quan="{$item/@quantity}"/>
```

Can we re-write the query without the where clause?

Example 9

<item num="557" name="Fleece Pullover" quan="1"/>,
<item num="563" name="Floppy Sun Hat" quan="1"/>,
<item num="443" name="Deluxe Travel Bag" quan="2"/>,
<item num="784" name="Cotton Dress Shirt" quan="1"/>,
<item num="784" name="Cotton Dress Shirt" quan="1"/>,
<item num="557" name="Fleece Pullover" quan="1"/>

Example 10

```
(: same join, no where clause :)

for $item in doc("order.xml")//item,
    $product in doc("catalog.xml")//product[number = $item/@num]
return <item num="{$item/@num}"
                name="{$product/name}“
                quan="{$item/@quantity}"/>
```

Example 11

```
(: order! :)

for $item in doc("order.xml")//item
order by $item/@num
return $item

for $item in doc("order.xml")//item
order by $item/@dept, $item/@num
return $item
```

Example 11

<item dept="ACC" num="443" quantity="2"/>,

<item dept="WMN" num="557" quantity="1" color="black"/>,

<item dept="WMN" num="557" quantity="1" color="navy"/>,

<item dept="ACC" num="563" quantity="1"/>,

<item dept="MEN" num="784" quantity="1" color="white"/>,

<item dept="MEN" num="784" quantity="1" color="gray"/>

Example 11

<item dept="ACC" num="443" quantity="2"/>,

<item dept="ACC" num="563" quantity="1"/>,

<item dept="MEN" num="784" quantity="1" color="white"/>,

<item dept="MEN" num="784" quantity="1" color="gray"/>,

<item dept="WMN" num="557" quantity="1" color="black"/>,

<item dept="WMN" num="557" quantity="1" color="navy"/>

Example 12

```
(: what does this do? :)
for $d in distinct-values(doc("order.xml")//item/@dept)
let $items := doc("order.xml")//item[@dept = $d]
order by $d
return <department code="{$d}">{
        for $i in $items
        order by $i/@num
        return $i
        }</department>
```

Example 12

```
<department code="ACC">
  <item dept="ACC" num="443"  quantity="2"/>
  <item dept="ACC" num="563" quantity="1"/>
</department>,
<department code="MEN">
  <item dept="MEN" num="784" quantity="1" color="gray"/>
  <item dept="MEN" num="784" quantity="1" color="white"/>
</department>,
<department code="WMN">
  <item dept="WMN" num="557" quantity="1" color="black"/>
  <item dept="WMN" num="557" quantity="1" color="navy"/>
</department>
```

Example 13

```
(: aggregate by group :)

for $d in distinct-values(doc("order.xml")//item/@dept)
let $items := doc("order.xml")//item[@dept = $d]
order by $d
return <department code="{$d}"
        numItems="{count($items)}"
        distinctItemNums="{count(distinct-values($items/@num))}"
        totQuant="{sum($items/@quantity)}"/>
```

Example 13

<department code="ACC" numItems="2" distinctItemNums="2" totQuant="3"/>,
<department code="MEN" numItems="2" distinctItemNums="1" totQuant="2"/>,
<department code="WMN" numItems="2" distinctItemNums="1" totQuant="2"/>

Extras

Enclosed expressions that evaluate to attributes

```
for $prod in doc("catalog.xml")/catalog/product
return <li>{$prod/@dept}number: {$prod/number}</li>
```

```
<li dept="WMN">number: <number>557</number></li>,
<li dept="ACC">number: <number>563</number></li>,
<li dept="ACC">number: <number>443</number></li>,
<li dept="MEN">number: <number>784</number></li>
```

Multiple conditions in where clause

```
for $prod in doc("catalog.xml")//product
let $prodDept := $prod/@dept
where $prod/number > 100
        and starts-with($prod/name, "F")
        and exists($prod/colorChoices)
        and ($prodDept = "ACC" or $prodDept = "WMN")
return $prod
```

Using an empty order declaration

```
declare default order empty greatest;
for $item in doc("order.xml")//item
order by $item/@color
return $item
```

```
<item dept="WMN" num="557" quantity="1" color="black"/>,
<item dept="MEN" num="784" quantity="1" color="gray"/>,
<item dept="WMN" num="557" quantity="1" color="navy"/>,
<item dept="MEN" num="784" quantity="1" color="white"/>,
<item dept="ACC" num="443" quantity="2"/>,
<item dept="ACC" num="563" quantity="1"/>
```

XML File – prices.xml

```
<prices>
  <priceList effDate="2006-11-15">
    <prod num="557">
      <price currency="USD">29.99</price>
      <discount type="CLR">10.00</discount>
    </prod>
    <prod num="563">
      <price currency="USD">69.99</price>
    </prod>
    <prod num="443">
      <price currency="USD">39.99</price>
      <discount type="CLR">3.99</discount>
    </prod>
  </priceList>
</prices>
```

Three-way join in a where clause

```
for $item in doc("order.xml")//item,
        $product in doc("catalog.xml")//product,
        $price in doc("prices.xml")//prices/priceList/prod
where $item/@num = $product/number
        and $product/number = $price/@num
return <item num="{$item/@num}"
        name="{$product/name}"
        price="{$price/price}"/>
```