

XML Data Management @ unibz - 2010/2011
Chris Mair

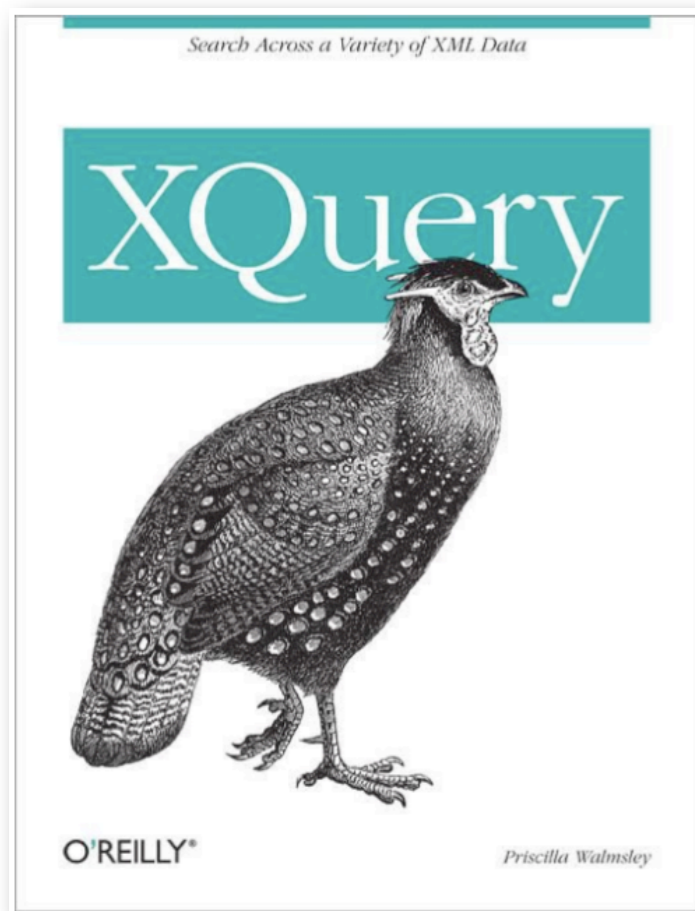
XQuery

XQuery

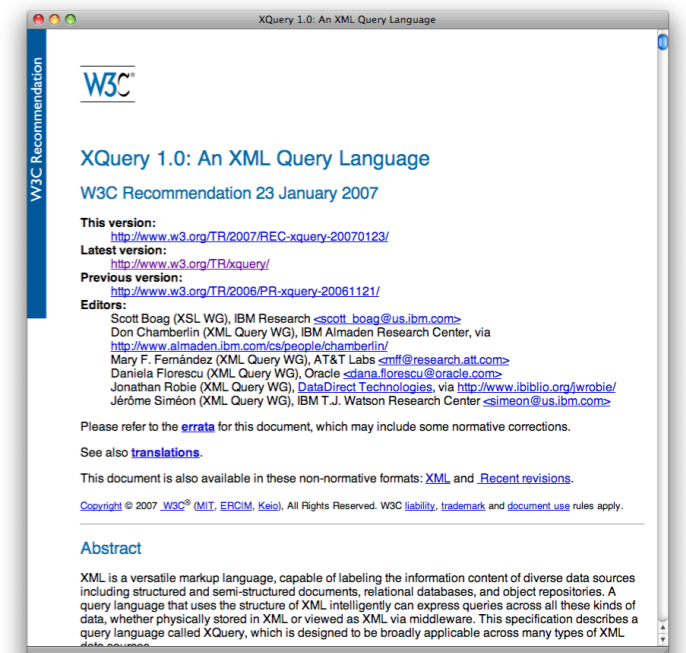
- is a W3C standard to query XML documents
- selects information from XML documents (stored in files or relational databases) that satisfies certain criteria
- reorganizes and transforms the information
- returns the information in an arbitrary structure

References

(if you wish to know the gory details)



XQuery
by Priscilla Walmsley,
O'Reilly 2007



<http://www.w3.org/TR/xquery/>

Running XQuery(s)

- XQilla is a compact, Open Source Implementation of XQuery 1.0 (and more)
- it has a cool logo ;)
- if `q1` is a file containing XQuery run it simply as

```
xqilla q1
```

- XQilla is available on xml.1006.org



Path Expressions

- Write Path expressions using XPath
- Example (**q1**):

```
doc("nobel.xml")/nobel/row/winner[text()='Marie Curie']
```

- *yields:*

```
<winner>Marie Curie</winner>
```

```
<winner>Marie Curie</winner>
```

FLWOR Expressions

(“flower”)

- Manipulate, transform and sort your results
- Example - see catalog.xml from the book (q2):

```
for $prod in doc("catalog.xml")/catalog/product
  where $prod/@dept = "ACC"
  order by $prod/name
  return $prod/name
```

- yields:

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

The **l**et clause

- Manipulate, transform and sort your results
- Example (q3):

```
for $product in doc("catalog.xml")/catalog/product
  let $name := $product/name
  where $product/@dept = "ACC"
  order by $name
  return $name
```

- still yields:

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

Constructing elements

(compare this to XSLT!)

- Example (q4):

```
<body>
  <h1>Physics</h1>
  <ul>{
    for $row in doc("nobel.xml")/nobel/row
      let $name := $row/winner/text()
      let $year := $row/yr/text()
      where $row/subject/text()="Physics"
      order by $year
      return <li>{$name} ({$year})</li>
  }</ul>
</body>
```

Joining sources

- Example (q5):

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

- yields:

```
<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" />
```

Grouping

(a.k.a. “dinner at the Curie’s”)

- Trick: distinct-values function, example (q6):

```
for $r in distinct-values(doc("nobel.xml")/nobel/row/winner)
  let $cnt := count(doc("nobel.xml")/nobel/row/winner[text()=$r])
  where $cnt > 1
  return <winner cnt="{ $cnt }">{$r}</winner>
```

- yields:

```
<winner cnt="2">Marie Curie</winner>
<winner cnt="2">Linus Pauling</winner>
<winner cnt="2">Frederick Sanger</winner>
<winner cnt="3">International Committee of the Red Cross</winner>
<winner cnt="2">Office of the United Nations High Commissioner for Refugees</winner>
<winner cnt="2">John Bardeen</winner>
```

Grouping: XQuery vs. SQL

(properly: Xqilla vs PostgreSQL)

- **q6**: 4.8 sec (on my notebook)
- can we make sure nobel.xml is read **only once**?

- yes (**q7**):

```
for $r in distinct-values(/nobel/row/winner)
  let $cnt := count(/nobel/row/winner[text()=$r])
  where $cnt > 1
  return <winner cnt="{ $cnt }">{$r}</winner>
```

- call this as **xqilla -i nobel.xml q7**
- **q7**: still 4.8 sec (so this **was** already the case)

Grouping: XQuery vs. SQL (properly: Xqilla vs PostgreSQL)

- oops: this is 1000 times faster in (Postgre)SQL:

```
chris=# \timing
```

```
Timing is on.
```

```
chris=# select winner from nobel group by winner having  
count(*) > 1;
```

```
winner
```

```
-----  
International Committee of the Red Cross  
Frederick Sanger  
John Bardeen  
Linus Pauling Marie Curie  
Office of the United Nations High Commissioner for Refugees  
(6 rows)
```

```
Time: 3.397 ms
```