University of Washington
Department of Computer Science & Engineering

Managing XML and Semistructured Data

Lecture 7: Query Languages - XQuery

Prof. Dan Suciu

Spring 2001

In this lecture

- Summary of XQuery
  - FLWR expressions
  - FOR and LET expressions
  - Collections and sorting

Resources

XQuery: A Query Language for XML Chamberlin, Florescu, et al.

W3C recommendation: www.w3.org/TR/xquery/

XQuery

- Based on Quilt (which is based on XML-QL)
- http://www.w3.org/TR/xquery/ 2/2001
- XML Query data model (of course ☺)
  - Ordered !

FLWR ("Flower") Expressions

FOR ... LET... FOR... LET...

WHERE...

RETURN...

XQuery

Find all book titles published after 1995:

```
FOR $x IN document("bib.xml")/bib/book
WHERE $x/year > 1995
RETURN $x/title
```

Result:

```
<title> abc </title>
<title> def </title>
<title> ghi </title>
```

XQuery

For each author of a book by Morgan Kaufmann, list all books she published:

```
FOR $a IN distinct(document("bib.xml")
                   /bib/book[publisher="Morgan Kaufmann"]/author)
RETURN <result>
          $a,
          FOR $t IN /bib/book[author=$a]/title
          RETURN $t
       </result>
```

distinct = a function that eliminates duplicates

XQuery

Result:

```
<result>
    <author>Jones</author>
    <title> abc </title>
    <title> def </title>
</result>
<result>
    <author> Smith </author>
    <title> ghi </title>
</result>
```

XQuery

- FOR $x in expr -- binds $x to each element in the list expr
- LET $x = expr -- binds $x to the entire list expr
  - Useful for common subexpressions and for aggregations

XQuery

```
<big_publishers>
    FOR $p IN distinct(document("bib.xml")//publisher)
    LET $b := document("bib.xml")/book[publisher = $p]
    WHERE count($b) > 100
    RETURN $p
</big_publishers>
```

count = a (aggregate) function that returns the number of elms

XQuery

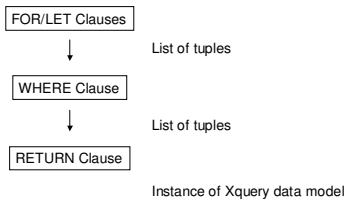
Find books whose price is larger than average:

```
LET $a=avg(document("bib.xml")/bib/book/@price)
FOR $b in document("bib.xml")/bib/book
WHERE $b/@price > $a
RETURN $b
```

XQuery

Summary:

- FOR-LET-WHERE-RETURN = FLWR



FOR v.s. LET

FOR

- Binds *node variables* → iteration

LET

- Binds *collection variables* → one value

FOR v.s. LET

```
FOR $x IN document("bib.xml")/bib/book
RETURN <result> $x </result>
```

Returns:

```
<result> <book>...</book></result>
<result> <book>...</book></result>
<result> <book>...</book></result>
...
```

```
LET $x := document("bib.xml")/bib/book
RETURN <result> $x </result>
```

Returns:

```
<result> <book>...</book>
         <book>...</book>
         <book>...</book>
         ...
</result>
```

Collections in XQuery

- Ordered and unordered collections
  - /bib/book/author = an ordered collection
  - Distinct(/bib/book/author) = an unordered collection
- LET $a = /bib/book → $a is a collection
- $b/author → a collection (several authors...)

```
RETURN <result> $b/author </result>
```

Returns:

```
<result> <author>...</author>
         <author>...</author>
         <author>...</author>
         ...
</result>
```

Collections in XQuery

What about collections in expressions ?

- $b/@price → list of n prices
- $b/@price * 0.7 → list of n numbers
- $b/@price * $b/@quantity → list of n x m numbers ??
- $b/@price * ($b/@quant1 + $b/@quant2) ≠ $b/@price * $b/@quant1 + $b/@price * $b/@quant2 !!

Sorting in XQuery

```
<publisher_list>
  FOR $p IN distinct(document("bib.xml")//publisher)
  RETURN <publisher> <name> $p/text() </name> ,
           FOR $b IN document("bib.xml")//book[publisher = $p]
           RETURN <book>
                    $b/title ,
                    $b/@price
                  </book> SORTBY(price DESCENDING)
         </publisher> SORTBY(name)
</publisher_list>
```

Sorting in XQuery

- Sorting arugments: refer to the name space of the RETURN clause, not the FOR clause
- To sort on an element you don't want to display, first return it, then remove it with an additional query.

If-Then-Else

```
FOR $h IN //holding
RETURN <holding>
         $h/title,
         IF $h/@type = "Journal"
           THEN $h/editor
         ELSE $h/author
       </holding> SORTBY (title)
```

Existential Quantifiers

```
FOR $b IN //book
WHERE SOME $p IN $b//para SATISFIES
  contains($p, "sailing")
  AND contains($p, "windsurfing")
RETURN $b/title
```

Universal Quantifiers

```
FOR $b IN //book
WHERE EVERY $p IN $b//para SATISFIES
  contains($p, "sailing")
RETURN $b/title
```

Other Stuff in XQuery

- BEFORE and AFTER
  - for dealing with order in the input
- FILTER
  - deletes some edges in the result tree
- Recursive functions
  - Currently: arbitrary recursion
  - Perhaps more restrictions in the future ?

Group-By in Xquery ??

- No GROUPBY currently in XQuery
- A recent proposal (next)
  - What do YOU think ?

Group-By in Xquery ??

```
FOR $b IN document("http://www.bn.com")/bib/book,
      $y IN $b/@year
WHERE $b/publisher="Morgan Kaufmann"
RETURN    GROUPBY $y
            WHERE count($b) > 10
            IN <year> $y </year>
```

← with GROUPBY

Equivalent SQL →

```
SELECT year
FROM Bib
WHERE Bib.publisher="Morgan Kaufmann"
GROUPBY year
HAVING count(*) > 10
```

Group-By in Xquery ??

```
FOR $b IN document("http://www.bn.com")/bib/book,
      $a IN $b/author,
      $y IN $b/@year
RETURN  GROUPBY $a, $y
          IN <result> $a,
                      <year> $y </year>,
                      <total> count($b) </total>
          </result>
```

← with GROUPBY

Without GROUPBY →

```
FOR $Tup IN distinct(FOR $b IN document("http://www.bn.com")/bib,
                       $a IN $b/author,
                       $y IN $b/@year
                     RETURN <Tup> <a> $a </a> <y> $y </y> </Tup>),
      $a IN $Tup/a/node(),
      $y IN $Tup/y/node()
LET $b = document("http://www.bn.com")/bib/book[author=$a,@year=$y]
RETURN <result> $a,
                      <year> $y </year>,
                      <total> count($b) </total>
          </result>
```

Group-By in Xquery ??

```
FOR $b IN document("http://www.bn.com")/bib/book,
      $a IN $b/author,
      $y IN $b/@year,
      $t IN $b/title,
      $p IN $b/publisher
RETURN
   GROUPBY $p, $y
   IN <result> $p,
            <year> $y </year>,
            GROUPBY $a
             IN <authorEntry>
                     $a,
                     GROUPBY $t
                     IN $t
                <authorEntry>
      </result>
```

← Nested GROUPBY's