Universität Freiburg
Institut für Informatik

Georges-Köhler Allee, Geb. 51
D-79110 Freiburg

Fang Wei-Kleiner

fwei@informatik.uni-freiburg.de

Advanced Databases and Information Systems
Summerterm 2019
Discussion on 28/05/2020

2. Sheet: XPath & XQuery

Exercise 1 (Evaluation of XQuery expressions)
You are given the following XML document, referred to as "book.xml":

```
<book>
  <title>Data on the Web</title>
  <author>Serge Abiteboul</author>
  <author>Peter Buneman</author>
  <author>Dan Suciu</author>
  <section id="intro" difficulty="easy" >
    <title>Introduction</title>
    <p>T1</p>
    <section>
      <title>Audience</title>
      <p>T1</p>
    </section>
    <section>
      <title>Web Data and the Two Cultures</title>
      <p>T2</p>
      <figure height="400" width="400">
        <title>Traditional client/server architecture</title>
        <image source="csarch.gif"/>
      </figure>
      <p>T2</p>
    </section>
  </section>
  <section id="syntax" difficulty="medium">
    <title>A Syntax For Data</title>
    <p>T1</p>
    <figure height="200" width="500">
      <title>Graph representations of structures</title>
      <image source="graphs.gif"/>
    </figure>
    <p>T1</p>
    <section>
      <title>Base Types</title>
      <p>T1</p>
    </section>
  </section>
</book>
```

a) Return the results for the following XQuery expressions. Explain in detail what the expressions are calculating.

(a)
```
<result> {
  for $x1 in doc("book.xml")//p
  where $x1/text()="T2"
  return
    for $x2 in doc("book.xml")//section
    where (some $x3 in $x2//p satisfies $x3/text()=$x1/text())
    return <x/>
} </result>
```
Ergebnis: `<result><x/><x/><x/><x/></result>`

Überprüfe für jeden Paragraphen (<p>) mit (möglicherweise zusammengesetzem) text()-Wert "T2" alle Sections (<section>) im Dokument. Wenn unterhalb einer solchen Section ein Paragraph mit einem text()-Wert identisch zum text()-Wert des ursprünglichen Paragraphen auftritt, wird der bachelor tag <x/> ausgegeben.
Es gibt zwei Paragraphen mit text()-Wert "T2". Diese treten in der selben Section auf, die jedoch (rekursiv) wiederum Teil einer Section ist. Diese beiden Sections haben also einen descendant-Paragraphen mit text()-Wert "2". Der bachelor tag <x/> wird also 2x2=4 mal ausgegeben.

(b)
```
<results> {
  for $x1 in doc("book.xml")//section
  where $x1//p/text()="T2"
  return
    <result> {
      for $x2 in $x1//p return $x2
    } </result>
} </results>
```
Ergebnis:
```
<results>
   <result>
      <p>T1</p>
      <p>T1</p>
      <p>T2</p>
      <p>T2</p>
   </result>
   <result>
      <p>T2</p>
      <p>T2</p>
   </result>
</results>
```

Das Ergebnis in der *return*-Klausel wird für jede Section berechnet, die einen Paragraphen mit text()="T2" besitzt (also insgesamt zwei mal). Das Ergebnis besteht dann jeweils aus einer Liste aller Paragraphen die innerhalb der entsprechenden Section auftreten.

Exercise 2 (Specifying XQuery expressions)
You are given XML document "bib.xml" which can be downloaded from ILIAS site under EX2. At the Oracle server (https://dbissql.informatik.uni-freiburg.de/dbis/dpod/sql.php) the XML file is stored in the table BIB. Specify XQuery expressions for the following goals.

a) Return all books for which the lastname of one of the authors matches with the lastname of the publisher. Assume that a publisher is always referred to by their lastname.
```
<books>{
  for $book in doc("bib.xml")/bib/book
  where $book/author/last = $book/publisher
  return $book
}</books>
```

b) For each book of author Peter Buneman, return the title and number of authors. If the book price is above 20 then also return the former.

```
<books>{
  for $book in doc("bib.xml")/bib/book
  where ($book/author/first/text() = "Peter" and
         $book/author/last/text() = "Buneman")
  return
    <book>
      { $book/title }
      <nrauthors>{ count($book/author) }</nrauthors>
      {
        if ($book/price/text() > 20)
        then $book/price
        else ()
      }
    </book>
}</books>
```

c) Return all pairs of different books of a publisher. The result must not contain duplcates. You can, however, assume that all book titles are different.

```
<bookpairs>{
  for $book1 in doc("bib.xml")/bib/book,
      $book2 in $book1/following::book
  where $book1/publisher=$book2/publisher
  return
    <pair>{ $book1, $book2 }</pair>
}</bookpairs>
```

d) For each author, return the lastname, name and the sum of all their (co-authored) books' prices.

```
<authors>{
  for $authors in fn:distinct-values(doc("bib.xml")/bib/book/author)
  let $author := doc("bib.xml")/bib/book/author[self::node()=$authors]
  let $book := doc("bib.xml")/bib/book[author=$authors]
  return
    <item>
      { $author[1]/last }
      { $author[1]/first }
      <pricesum>{ fn:sum($book/price) }</pricesum>
    </item>
}</authors>
```

e) For each author, return a list of his/or book titles (ordered by price). An element of the list has to contain name and lastname of an author, and constrained to three books. In addition, add a tag "minprice" with the price of the cheapest book of the author.

```
<authors>{
  for $authors in fn:distinct-values(doc("bib.xml")//author)
  let $author := doc("bib.xml")/bib/book/author[self::node()=$authors]
  let $book :=
    for $book2 in doc("bib.xml")/bib/book[author=$authors]
    order by $book2/price/text()
    return $book2
  return
    <item>
      { $author[1] }
      { $book[position()<=3]/title }
      <minprice>{ fn:min($book/price/text()) }</minprice>
    </item>
}</authors>
```