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 02/05/2019

1. Sheet: XML & XPath

Exercise 0 (Setup)

For learning how to construct queries for travsering XML as well as JSON documents, we suggest to work with our test workbench for SQL (`https://dbissql.informatik.uni-freiburg.de/dbis/dpod/sql.php`) as a graphical interface to Oracle databases. You have to login with your student credentials and then choose the `mondial_xml_pph` database. Alternatively, you could download the `mondla.xml` file from ILIAS and directly access it via any programming library for XML which supports XPath.

Exercise 1 (XPath Queries)

Use *mondial.xml* to answer the following questions with XPath.

a) What are the names of the countries with more than 10 million citizens and total area less than 200000 km^2?

b) What are the names of countries which have a smaller area than the Netherlands?

c) What are the names of the countries which share a border with Germany while having a higher population growth than Germany?

d) What are the names of the capitals which are situated at at least one waterside?

e) What are the names of all cities which are situated at a lake?

f) What are the names of all rivers where at least one capital is situated at?

g) What are all "German leaf nodes"? More specifically, what are all elements in Mondial which are located in the subtree of a `country` element with `car_code=''D''` and do not have any children themselves?

Exercise 2 (XPath - Axes und Equivalence)

a) You are given the following XPath request to compare it with XPath requests 1 to 4:

```
//n[preceding-sibling::n]
```

Specify XML documents for each of the following XPath requests, such that the respective two requests return **different** results.

   (a) `//n[preceding::n]`
   (b) `//n[preceding::n and following-sibling::n]`
   (c) `//n[preceding::n and parent::*/child:n]`
   (d) `//n/preceding::n[following-sibling::*]`

b) Find a XPath request which is equivalent to `//n[preceding-sibling::n]`, while not using `preceding-sibling`.

c) You are given the following XPath request to compare it with XPath requests (1) to (3):

`//n[parent::n and child::n].`

Specify XML documents for each of the following XPath requests, such that the respective two requests return **different** results.

(a) `//n[ancestor::n and child::n]`

(b) `//n[child::n/child::n]/child::n`

(c) `//n[preceding::n and parent::*/child::n]`

d) Find a XPath request which is equivalent to `//n[parent::n and child::n]`, while not using `parent` or "`..`".

e) Let p be a node in a XML-tree. Specify a XML request to return the set of all nodes of the XML-tree which are **different** than p.

f) You are given the following two XPath requests:

```
//City[preceding::City[1]/CName = "Freiburg"]/CName
//City[(preceding::City)[1]/CName = "Freiburg"]/CName
```

Specify equivalent requests **without** using backward-axes.

Exercise 3 (XPath & XRel)

You are given the XML document "bib.xml"[1], containing a structured bibliography. For each of the following queries, give the XPath expression that answers the query.

XRel [2] is an alternative approach to store and retrieve XML documents via relational databases. Write the corresponding SQL queries using XRel after formulating the needed tables.

a) Output all unique authors' lastnames.

b) Output all the books[3] published by "Addison-Wesley".

c) Output all the names of books that was published after the year 1994 with a price lower than 50.

[1] `http://tinyurl.com/small-bib-xml`

[2] `https://dl.acm.org/citation.cfm?id=383038`

[3] It is enough to output the starting position and the end position of the corresponding element instead of returning the element it self.