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 2020

Discussion on 19/05/2020

# 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$^2$?

```
/mondial/country[population >10000000 and @area<200000]/name
```

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

```
/mondial/country[@area < //country[@car_code=’NL’]/@area]/name
```

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

```
//country[border/@country=’D’ and (./population_growth > /mondial/country[@car_code=’D’]/
population_growth/text())]/name
```

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

```
/mondial/country/((province/city)|city)[@id=/mondial/country/@capital and located_at/@watertype]/name
```

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

```
/mondial/country/(city|province/city)[located_at/@lake]/name/text()
```

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

```
/mondial/river[@id = /mondial/country(city|(province/city))[@id = /mondial/country/@capital]
/located_at/@river]
```

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?

```
/mondial/country[@car_code="D"]/*[(not (./*))]
```

### 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::*]`

Dieses Dokument gilt für Teilaufgaben (1)-(4).

```
<a>
  <b>
    <n id="1"/>
    <c/>
  </b>
  <n id="2"/>
  <n id="3"/>
</a>
```

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

Eine mögliche Lösung ist die Anfrage `//n/following-sibling::n`.

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]`

Vorschlag zu Teilaufgabe (1):

```
<n id="1">
  <a>
    <n id="2">
      <n id="3"/>
    </n>
  </a>
</n>
```

Vorschlag zu Teilaufgaben (2) und (3):

```
<n id="1">
  <n id="2">
    <n id="3"/>
  </n>
  <n id="4"/>
</n>
```

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

Eine mögliche Lösung ist die Anfrage `//n/n[child::n]`.

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 $p$.

`(p/preceding::node()|p/ancestor::node()|p/descendant::node()|p/following::node())`

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.

- `//City[CName = "Freiburg"]/following::City[1]/CName`

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

### 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.

Element

| Start | End | PId |
|---|---|---|
| 0 | 1058 | 1 |
| 5 | 167 | 2 |
| 23 | 48 | 4 |
| 56 | 101 | 5 |
| 64 | 76 | 6 |
| 84 | 93 | 7 |
| 110 | 135 | 8 |
| 147 | 159 | 9 |
| 174 | 415 | 2 |
| 192 | 243 | 4 |
| 251 | 296 | 5 |
| 259 | 272 | 6 |
| 279 | 289 | 7 |
| 305 | 349 | 5 |
| 313 | 324 | 6 |
| 331 | 341 | 7 |
| 358 | 383 | 8 |
| 359 | 407 | 9 |
| 422 | 798 | 2 |
| 440 | 462 | 4 |
| 470 | 520 | 5 |
| 478 | 493 | 6 |
| 500 | 512 | 7 |
| 529 | 577 | 5 |
| 537 | 550 | 6 |
| 557 | 569 | 7 |
| 586 | 630 | 5 |
| 594 | 605 | 6 |
| 612 | 622 | 7 |
| 639 | 720 | 10 |
| 647 | 662 | 11 |
| 669 | 681 | 12 |
| 689 | 706 | 13 |
| 729 | 766 | 8 |
| 778 | 790 | 9 |
| 805 | 1050 | 2 |
| 823 | 884 | 4 |
| 892 | 971 | 10 |
| 900 | 913 | 11 |
| 920 | 932 | 12 |
| 940 | 957 | 13 |
| 980 | 1017 | 8 |
| 1029 | 1042 | 9 |

Attribut

| PId | Start | End | Value |
|---|---|---|---|
| 3 | 6 | 6 | 1994 |
| 3 | 175 | 175 | 1992 |
| 3 | 423 | 423 | 2000 |
| 3 | 806 | 806 | 1999 |

Text

| Start | End | Value | PId |
|---|---|---|---|
| 30 | 47 | TCP/IP Illustrated | 4 |
| 70 | 76 | Stevens | 6 |
| 91 | 92 | W. | 7 |
| 121 | 134 | Addison-Wesley | 8 |
| 154 | 158 | 65.95 | 9 |
| 199 | 242 | Advanced Programming in the Unix environment | 4 |
| 265 | 271 | Stevens | 6 |
| 286 | 287 | W. | 7 |
| 319 | 323 | Suciu | 6 |
| 338 | 340 | Dan | 7 |
| 369 | 382 | Addison-Wesley | 8 |
| 402 | 406 | 65.95 | 9 |
| 447 | 461 | Data on the Web | 4 |
| 481 | 492 | Abiteboul | 6 |
| 507 | 511 | Serge | 7 |
| 543 | 549 | Buneman | 6 |
| 564 | 568 | Peter | 7 |
| 600 | 604 | Suciu | 6 |
| 619 | 621 | Dan | 7 |
| 653 | 661 | Abiteboul | 11 |
| 676 | 680 | Serge | 12 |
| 702 | 705 | CITI | 13 |
| 740 | 765 | Morgan Kaufmann Publishers | 8 |
| 785 | 789 | 39.95 | 62 |
| 830 | 883 | The Economics of Technology and Content for Digital TV | 4 |
| 906 | 912 | Gerbarg | 11 |
| 927 | 931 | Darcy | 12 |
| 953 | 956 | CITI | 13 |
| 991 | 1016 | Kluwer Academic Publishers | 8 |
| 1036 | 1041 | 129.95 | 9 |

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

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

Path

| PId | PathExpr |
|---|---|
| 1 | #/bib |
| 2 | #/bib#/book |
| 3 | #/bib#/book#/@year |
| 4 | #/bib#/book#/title |
| 5 | #/bib#/book#/author |
| 6 | #/bib#/book#/author#/last |
| 7 | #/bib#/book#/author#/first |
| 8 | #/bib#/book#/publisher |
| 9 | #/bib#/book#/price |
| 10 | #/bib#/book#/editor |
| 11 | #/bib#/book#/editor#/last |
| 12 | #/bib#/book#/editor#/first |
| 13 | #/bib#/book#/editor#/affiliation |

a) Output all unique authors' lastnames.

```
/distinct-values(//author/last)
SELECT distinct Text.Value
FROOM  Text T, Path P
WHERE P.PathExpr like '#%/author#/last'
   AND T.PId = P.PId
```

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

```
//book[./publisher = "Addison-Wesley"]
SELECT E1.Start, E1.End
FROM Element E1, Element E2, Text T, Path P1, Path P2
WHERE P1.PathExpr like '#%/book'
   AND E1.pId = P1.PId
   AND P2.PathExpr like '#%/book#/publisher'
   AND E2.pId = P2.PId
   AND T.Value = "Addison-Wesley"
   AND T.Start > E2.Start
   AND T.End < E2.End
   AND E1.Start < E2.Start
   AND E1.End > E2.End
```

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

```
//book[./@year > 1994 and ./price < 50]
```

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