Prof. Dr. Stefan Brass
Institut für Informatik
MLU Halle-Wittenberg

December 8, 2017

XML and Databases
— Exercise Sheet 8 —

You only have to submit the parts marked as “Homework Exercises”, i.e. Part d). But please think about the questions in Part a) before the meeting! Send your homework solutions to the instructor via EMail: `brass@informatik.uni-halle.de` (with “`xml17`” in the subject line). The official deadline is December 14, 10:00 (before the lecture time).

Repetition Questions

a) Answer the following questions about XPath:

- XPath has no reserved words, i.e. all valid XML identifiers can be used as element names (no special escaping of certain element names is needed). Give some examples, how keywords in XPath can be distinguished from element names.
- What does the XPath-Expression “`x-1`” mean?
- How are variables written in XPath?
- If one writes a string constant in XPath with the delimiter `'`, how can one include this character as data in the string? Why does the XML solution “`'`” not work in XPath? (Think of XPath expressions contained in XML attributes, as in XSLT stylesheets. This also explains why the situation that you need the delimiter itself in the string constant is not so seldom: You cannot use the other delimiter `"`, because that is already taken for delimiting the attribute value.)
- How can you write a date value in XPath, e.g. December 6, 2017?
- How can you write boolean values in XPath?
- How can you access the three main components of the context in XPath (context item, context position, and context size)?
- What does the comma operator “`,`” do? Compare it with the union operator “`|`” (also written “`union`”).
- Explain atomization. How is the function called that does atomization and where is it done implicitly? How is the result of atomizing an input sequence defined? In which situation atomization can give an error?
- How are the value comparison operators written in XPath? What is the main difference to the general comparison operators?

- Explain why a condition like “`@POINTS ge 8`” can give a type error for documents that were not validated (even if the attribute value would be a legal integer constant). How can one solve the problem?
- Given an example where `$x = 1` and `$x != 1` hold at the same time.
- Give an example where “`=`” violates the transitivity axiom.
- How can one check in XPath whether two nodes are the same? How can one check whether a node appears before another node (in document order)?
- How is universal and existential quantification written in XPath? Explain syntax and semantics. Why is it a problem if the condition generates a runtime error for some, but not all of the values inserted for the variable?
- Explain syntax and semantics of `for`-expressions in XPath.
- Normally, loops are typical constructs of imperative languages. Compare `for`-loops in a language like Java with `for`-expressions in XPath, and explain why XPath is not imperative. How might `for`-expressions in XPath be evaluated on parallel hardware? Why are typical loops in Java much harder to parallelize?
- Explain the syntax and semantics of `if`-expressions in XPath. Why is there no version without `else`? Compare it with `if`-statements in Java.

In-Class Exercises

b) Consider the following XML document:

```
<?xml version="1.0"?>
<BOOKLIST>
    <BOOK ISBN="0-13-014714-1" PAGES="1074">
        <AUTHOR FIRST="Paul" LAST="Prescod"/>
        <AUTHOR FIRST="Charles" LAST="Goldfarb"/>
        <TITLE>The XML Handbook - 2nd Edition</TITLE>
        <PUBL DATE="19991112">Prentice Hall</PUBL>
        <NOTE>Contains CD.</NOTE>
    </BOOK>
    <BOOK ISBN="1-56592-709-5" PAGES="107">
        <AUTHOR FIRST="Robert" LAST="Eckstein"/>
        <TITLE>XML Pocket Reference</TITLE>
        <PUBL DATE="19991001">O'Reilly</PUBL>
    </BOOK>
</BOOKLIST>
```

It can be downloaded from this web address:

[http://users.informatik.uni-halle.de/~brass/xml17/booklist.xml]

- What is the full version of the following expression?

```
/*//AUTHOR/@LAST
```

- Please write an XPath expression to print the last names of all authors.

  Assume that the context node is the document node (i.e. the root node) and that it suffices to select the attribute nodes, and not necessarily take their value. E.g. this XSLT construct takes automatically the value of the attribute nodes:

```
<xsl:value-of select="..." separator=","/>
```

- What is the difference between the following XPath expressions?
  - `//TITLE`
  - `//TITLE/text()`
  - `data(//TITLE)`

c) Consider the homework grades databases with data in elements:

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<GRADES-DB>
    <STUDENT>
        <SID>101</SID>
        <FIRST>Ann</FIRST>
        <LAST>Smith</LAST>
    </STUDENT>
    ...
    <RESULT>
        <SID>101</SID>
        <CAT>H</CAT>
        <ENO>1</ENO>
        <POINTS>10</POINTS>
    </RESULT>
    ...
</GRADES-DB>
```

It is available at: [http://users.informatik.uni-halle.de/~brass/xml17/ex2.xml]

- Please write XPath query to find the SID of Ann Smith.

  It suffices that the SID element is selected. If necessary, one can explicitly call `data(...)` to perform an atomization.

- Print the last names of all students who got more than 8 points for Homework 1.

  This exercise already requires a (semi-)join. One can apply the `some`-quantifier to get a variable for one of the nodes, and use the context for the other node.

- What is the error in

```
//EXERCISE[some $r in //RESULT satisfies POINTS = MAXPT]
```


Homework Exercises

d) Please download again the XML file for the classical music CDs (there was a small problem with the English translation):

- Data file (this has changed):
  [http://www.informatik.uni-halle.de/~brass/xml17/cd.xml]
- XML Schema definition (this has changed):
  [http://www.informatik.uni-halle.de/~brass/xml17/cd.xsd]
- DTD:
  [http://www.informatik.uni-halle.de/~brass/xml17/cd.dtd]

The document has the following elements:

- `CDDB`: `(composers, cds?, soloists?)`
- `composers`: `(composer*)`
- `composer`: `(pieceOfMusic*)`, attributes: `cno`, `firstName`, `name`, `born`, `died`.
- `pieceOfMusic`: `(recording*)`, attributes: `pno`, `title`, `key`, `opus`.
- `recording`: empty content, attributes: `rno`, `orchestra`, `conductor`.
- `cds`: `(cd*)`.
- `cd`: `(track*)`, attributes: `cdno`, `name`, `producer`, `numDiscs`, `totalTime`.
- `track`: empty content, attribute: `rno`.
- `soloists`: `(soloist*)`.
- `soloist`: `(performance*)`, attribute: `name`.
- `performance`: empty content, attributes: `rno`, `instrument`.

Please write the following queries in XPath and test it with an actual XPath implementation:

- Print the names of all soloists that play violin. Since a soloist can play more than one instrument (e.g. violin and viola are quite common), the instrument is stored in the performance element nested in the soloist element.
- Print all CDs (element nodes) that contain more than three discs, i.e. have at least the value 4 in the attribute `numDiscs`.
- Which CDs (print the attribute `name`) contain compositions of "Wolfgang Amadeus Mozart"?
- Select all componists for which no opus numbers are used, i.e. none of their pieces of music contains the attribute "`opus`".