Lab 3 – DD1334 – XML, Xpath, Xquery

Important: You and your lab partner will be expected to be able to answer questions about your code and if you understand the results.
Goals: This lab focusses on some problems involving XML in a Data Science setting.

Before you start, please make sure you have read the relevant chapters of the book pertaining to XML/Xpath/Xquery that we have covered in the lectures.

Preparations:
We will work with the same fact based database as before, but this time using XML!

Download the sources:
https://www.dbis.informatik.uni-goettingen.de/Mondial/mondial.dtd
https://www.dbis.informatik.uni-goettingen.de/Mondial/mondial.xml
https://www.dbis.informatik.uni-goettingen.de/Mondial/mondial.xsd

If you work on u-shell, you can use the preinstalled program xqilla there, otherwise install it on your local system. Xqilla takes as input a Xquery file. Your answers should be saved as prepared query files for each question that you can demonstrate.

Example Xquery query.txt:

```
let $d:=doc("mondial.xml")
return $d/mondial/country/name
```

Example execution of xqilla in same directory as both query.xml and mondial.xml:

```
xqilla query.xml
<name>Albania</name>
<name>Greece</name>
<name>North Macedonia</name>
<name>Serbia</name>
<name>Montenegro</name>
<name>Kosovo</name>
<name>Andorra</name>
<name>France</name>
<name>Spain</name>
...
```

Questions

To develop an understanding of the data, start by reading the mondial.dtd file. Note that in all answers below, we do not care about whitespaces/newlines, so the results from xqilla may not be in exactly the same pretty formatting as shown. "..." indicates more data in the actual output.

a) Return a list of all mountains with type volcano. Result should look as follows:

```
<name>Hekla</name>
<name>Katla</name>
<name>Croscat</name>
<name>Vesuvio</name>
<name>Etna</name>
<name>Pico de Teide</name>
<name>Pico</name>
<name>Cabeço Gordo</name>
…
```

b) Return a list of all mountain names with type volcano that have a mountains tag indicating that they lie in Hawaii. Result looking as follows:

```
<name>Mauna Kea</name>
<name>Mauna Loa</name>
<name>Haleakala</name>
<name>Mauna Kamakou</name>
<name>Ka‘ala</name>
<name>Kawaikini</name>
<name>Lānaihale</name>
```

c) Return a list of mountains with height above 8000m formatted as follows and ordered by increasing height. Note, you need to convert height to float with xs:float(…).

```
<bigmountain><height>8027 meters</height><name>Shishapangma</name></bigmountain>
<bigmountain><height>8034 meters</height><name>Gasherbrum II</name></bigmountain>
<bigmountain><height>8051 meters</height><name>Broad Peak</name></bigmountain>
<bigmountain><height>8080 meters</height><name>Gasherbrum I</name></bigmountain>
<bigmountain><height>8091 meters</height><name>Annapurna</name></bigmountain>
<bigmountain><height>8125 meters</height><name>Nanga Parbat</name></bigmountain>
<bigmountain><height>8163 meters</height><name>Manaslu</name></bigmountain>
<bigmountain><height>8167 meters</height><name>Dhaulagiri</name></bigmountain>
<bigmountain><height>8188 meters</height><name>Cho Oyu</name></bigmountain>
<bigmountain><height>8485 meters</height><name>Makalu</name></bigmountain>
<bigmountain><height>8516 meters</height><name>Lhotse</name></bigmountain>
<bigmountain><height>8586 meters</height><name>Kangchendzonga</name></bigmountain>
<bigmountain><height>8611 meters</height><name>K2</name></bigmountain>
<bigmountain><height>8848 meters</height><name>Mt. Everest</name></bigmountain>
```

d) Note that cities may have multiple name variations. Return an XML document of the following form listing all cities per country as well as the various alias names. Hint: recall that Xquery results can be nested, you can use multiple variable declarations and remember that the result of "return" is again a sequence of items. The result can be assigned to a variable as desired.

```
<mylist>
        <country name="Albania">
                <city id="cty-Albania-Tirane">
                        <alias>Tirana</alias>
                        <alias>Tirane</alias>
                </city>
                <city id="stadt-Shkoder-AL-AL">
                        <alias>Shkodër</alias>
                </city>
                ...
        </country>
        <country name="Greece">
                <city id="cty-Greece-Komotini">
                ...
                </city>
                ...
        </country>
        …
</mylist>
```

e) Return the number of cities available per country for those countries with more than 40 cities. For countries with more than 60 cities in the database, we add the tag note="morethan60". The result should look like shown below. Note that the name attribute contains the name of the country.

```
<manycities><country name="France">41</country><country note="morethan60" name="Spain">65</country><country note="morethan60"
name="Germany">85</country><country name="Italy">56</country><country name="Poland">41</country><country note="morethan60"
name="Russia">171</country><country name="Romania">42</country><country note="morethan60" name="Turkey">88</country><country
note="morethan60" name="United Kingdom">84</country><country note="morethan60" name="China">302</country><country
name="Iran">50</country><country note="morethan60" name="India">99</country><country
name="Indonesia">58</country><country note="morethan60" name="Japan">72</country><country note="morethan60"
name="Mexico">83</country><country note="morethan60" name="United States">251</country><country name="Colombia">52</country><country
note="morethan60" name="Brazil">210</country><country name="Nigeria">58</country></manycities>
```

f) Consider the file newdata.xml. Use Xquery to return a new xml document in the same format as newdata.xml which also adds the older population counts from mondial.xml for the cities in newdata.xml.