

## Homework 5 (Upload to Canvas by 23:59:59 EDT on April 16)

Please submit the homework electronically under Canvas (under hw5). To understand and solve this assignment please read textbook sections 7.4, 27.6, 27.7 For more help you can refer to the W3Schools tutorials at <http://www.w3schools.com>.

The goal of this assignment is to learn more about DTDs, XSD, and practice writing queries in XPath and XQuery. You should start by downloading Zorba (<http://zorba.28.io/>), an XQuery engine which can be used through the command line or using an Eclipse plugin. To run your queries through the command line, this document is helpful:

<http://cf.zorba-xquery.com.s3.amazonaws.com/doc/zorba-1.2.0/zorba/html/commandline.html>

You are free to use any XQuery engine you like, but the instructors will be grading your assignment using Zorba.

In this assignment you are given the following:

- **tripster.xml**: This xml file contains a huge chunk of data, which you will use to execute your queries. The data contains information about users, their trips, their trip albums, comments, ratings and photos.
- **tripster.xsd**: This schema will help you in understanding the schema of the xml file above.
- **courses.dtd**: This DTD will be used in creating XML and XSD documents for question 2.

1. (15 points)

Analyze the XSD and XML file. Write a DTD describing the structure for the given XML file. You may omit the simple text data definitions (e.g. `<!ELEMENT name (#PCDATA)>`).

Submit your answers in a file `tripster.dtd`.


2. (20 points)


Using `courses.dtd`, create an XSD file and an XML file with at least 2 entries. Invent some interesting enumerated types for day and time of classes. Make sure your newly created XML, XSD is validated with respect to the given DTD.

Submit answers for the above as files `courses.xml` and `courses.xsd`

3. (10 points)

Update the existing schema definition (`tripster.xsd`) as follows, and validate that it is correct with respect to `tripster.xml`. You might find the following helpful for this:

<http://www.utilities-online.info/xsdvalidation/#.VR8FQUZAsd> 

- (a) Using the keys `userID` and `tripID` as a model, add a key `locationID` (which is its name) for `locationType`.
- (b) Add a key `contentID` for `contentType` (which is its id). 
- (c) Using `requestTripRef` as a model, add a `keyref dreamLocationRef` for `userType` such that the dream should be an existing location.
- (d) Add a `keyref inviteTripRef` for `userType` such that a friend is an existing user.

Submit answers for the above as a file `tripster-revised.xsd`

4. (5 points)

Write XPath expressions for the following:

- (a) The names (text) of all users.
- (b) The names (node subtrees) of all trips with `privacyFlag` set to "public".

Submit answers for the above as a file `hw5-4.pdf`

The next five exercises ask you to write queries in XQuery. Your answers will consist of the query itself (not the answer returned by the query). Please put each query in a separate file; name them hw5-5.xq ... hw5-9.xq, and use Canvas to submit them. Your script must compile and return the results using tripster.xml in the format given below. Your scores will be based on correctness, implementation and style. You are not allowed to hard code the answers to any of the xqueries.

5. (10 points)

For each user interested in hiking, return their name, email address and interest. You may use the string function "fn:contains(string1,string2)" which returns true if string1 contains string2. You should catch "hiking" and "Hiking". Your result should have the following form:

```
<results>
<user>
  <name> {name} </name>
  <email> {email} </email>
  <interest> {interests} </interest>
</user>
...
</results>
```

6. (10 points)

For each album, return its name and the total number of photos in the album. Your result should have the following form:

```
<results>
<album>
  <name> {name} </name>
  <photo_count> {total number of photos} </photo_count >
</album>
...
</results>
```

7. (10 points)

For each content, return its url, album name and the trip name. Your result should have the following form:

```
<results>
<content>
  <url> {url} </url>
  <album_name > {album name to which content belongs} </album_name>
  <trip_name> {trip name to which content belongs} </trip_name>
</content>
...
</results>
```

8. (10 points)

For each feature, return a list of trips with that feature. Your result should have the following form:

```
<results>
<feature>
  <name> {feature} </name>
```

```

    <list>
      <trip> {name} </trip>
      ...
    </list>
  </feature>
  ...
</results>

```

9. (10 points)

Return all the trips with their average rating score as well as all comments (if a comment is empty then omit the comment). If a trip has not been rated, then just return the name and omit the average score and comments. (**Hint:** You may find it helpful to read about XQuery Conditional Execution and use fn:empty.)

Your result should have the following form:

```

<results>
  <trip>
    <name> {trip-name} </name>
    <average_score> {average of all the scores} </average_score>
    <comments>
      <comment> {comment} </comment>
      ....
    </comments>
  </trip>
  ...
  <trip>
    <name> {trip-name} </name>
  </trip>
  ...
  ...
</results>

```