Database systems, winter 2015-2016

Dry homework#4

Submission until January 21\textsuperscript{th} at 13:30, pairs only, in the course’s cell on the first floor.

Teaching assistant in charge: Hadar Frenkel.

For questions please contact hfrenkel@cs.technion.ac.il with the subject HW4.

Note that there is also a wet part for this homework

Students’ details:

<table>
<thead>
<tr>
<th>name</th>
<th>id</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Please return to cell:________

Final grade:_______________
Question 1: Neo4j

You are given a graph database that contains nodes of the following types:

<table>
<thead>
<tr>
<th>reader</th>
<th>book</th>
<th>author</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>ISBN</td>
<td>id</td>
</tr>
<tr>
<td>name</td>
<td>title</td>
<td>name</td>
</tr>
<tr>
<td></td>
<td>library</td>
<td></td>
</tr>
</tbody>
</table>

With the following edges:

**Reads** is an edge between reader and book, and has the properties *date, how_long*

**Wrote** is an edge between author and book and has the property *publish_date*

a. Find the average length of reading books that were written by David Grossman.
   MATCH (:reader)-[r:Reads]->(:book)<-[w:Wrote](:author{name:"David Grossman"})
   RETURN avg(r.how_long)

b. Find all readers that read at least one book after the publish date of that book or read a book written by J.K Rowling. You can assume here that there is exactly one publication for each book.
   MATCH (re:reader)-[r:Reads]->(:book)<-[w:Wrote](a:author)
   WHERE ((r.date> w.publish_date) OR (a.name = "J.K Rowling"))
   RETURN DISTINCT re.name
c. Find all readers that their reader distance from the reader Danny is at most K.
Reader distance 1 means that two readers read a common book.
Reader distance 2 between reader R1 and reader R2 means that there exists books B1 and B2 such that R1 read B1 and R2 read B2 and there exists a reader R3 that read B1 and B2 (B1 might be the same as B2).
Reader distance K between reader A and reader B means that there exists a reader C such that his distance from A is k-1 and from B is 1.
Reader distance 0 is the reader itself.

MATCH (r:reader)-[:Reads*0..2k]-> (:reader{name:“Danny”})
RETURN DISTINCT r
a. Write a DTD that describes movie library, satisfies the following:
   - A movie has at least one producer, a title, at least one actor and a genre, which is one of: drama, comedy or science fiction.
   - An actor has a name, a unique ID and the IDs of actors the actor played with in movies.

```xml
<!ELEMENT movie_library (movie*)>
<!ELEMENT movie (producer+, title, actor+)>  
<!ATTLIST movie genre (drama | comedy | science fiction) #REQUIRED>
<!ELEMENT producer (#PCDATA)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT actor (name)>  
<!ATTLIST actor id ID #REQUIRED
actors IDREFS #IMPLIED>
<!ELEMENT name (#PCDATA)>
```

b. Given the DTD:
```xml
<!ELEMENT library (books, readers)>  
<!ELEMENT books (book)*>  
<!ELEMENT readers (reader)*>  
<!ELEMENT book (type?, year, author+,title)>  
<!ATTLIST book bid ID #REQUIRED>
<!ELEMENT type (#PCDATA)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT year (#PCDATA)>
<!ELEMENT author (#PCDATA)>
<!ELEMENT reader (name)>  
<!ELEMENT name (#PCDATA)>
<!ATTLIST reader rid ID #REQUIRED>
<!ATTLIST reader read_books IDREFS #REQUIRED>
```

1. Write an XML file that follows the given DTD. Each element in the DTD must appear in your file.

```xml
<?xml version="1.0"?>
<!DOCTYPE library SYSTEM "library.dtd">
<library>
  <books>
    <book bid="a1">
```
2. Write an XPath 2.0 query that returns for each reader its name and the title of the latest-published book that the reader read.

   for $r in //reader return
   ($r/name, //title[./../.=id($r/@read_books) and
   (every $b in //book[. = id($r/@read_books)] satisfies
   $b/year<= ./../year)])

3. Write an XPath 2.0 query that returns the libraries that own all of the books in the database, and has at least two readers in it.

   //library[(every $b in //book satisfies $b=./books/book) and
   count(.//readers/reader)>1]