University of Toronto
CSC**343, Fall 2013**

Assignment 4

Due date: Wed 4 Dec 2013 at 10:00pm

1 Introduction

Our Miniature Internet Movie Database (MIMDb) contains semi-structured data that holds information about Hollywood movies, actors, directors and Oscar awards. We are ultimately interested in designing a website that enables users to browse this data and extract useful information. You will be given the document type definitions (DTDs) of all the main entities in the MIMDb schema. Your job is to simply implement a list of XQueries that will extract specific information from the schema.

Additionally, you will be given some XML files to help you test the correctness of your queries. However, your queries should be designed to return correct answers on *any* XML data files that satisfy these DTDs. You are encouraged to extend the datasets test your queries on the extended datasets besides the ones provided for you here. The main entities in the MIMDb schema and the DTDs that describe them are as follows:

- **Movies.** In the file `movies.dtd` you will find the specification for XML data that hold information about movies, such as the year they were produced, the actors who appeared in them, or the Oscar awards they won (if any).
- **People.** In the file `people.dtd` you will find the specification for XML data that contain personal information about actors and directors, including their names, gender, date-of-birth, or any Oscar awards they might have won (if any).
- **Oscars.** In the file `oscars.dtd` you will find the specifications for XML data related to Oscar awards that have been awarded either to movies or individuals (actors/directors), including the year of the award and its type (i.e. name).

Notice the following supported features in the MIMDb schema:

- A person can be both an actor and a director (whether in the same movie or in different movies).
- An Oscar award can be associated with a person only, a movie only, or both a person and a movie (e.g., for a person's role in a certain movie).
- A movie can be directed by one director only, but will have one or more actors (no less than one actor).

2 Query XML Documents

Write queries in XQuery to produce the following results:

1. The average number of oscars won by any person (whether an actor or director).
2. For every movie in the database, the movie ID and the number of actors who acted in it.
3. Which directors do not have their place of birth (pob) listed in the database? Return their person ID (PID) and last name. Duplicates are allowed.
4. In which movies did Sam Worthington act? return the movie title and movie year. Order by movie year, in descending order.
5. All movies that were directed by James Cameron since the year 2001 (inclusive). Assume there is only one "Person" with the name James Cameron.
6. For any movie that has won one or more Oscar awards, return the movie title and the type of each award.
7. For each Oscar award type ever given to a *movie*, find in which year it was awarded for the first time. Return the name of the Oscar award (Type), the year it was first awarded, and the title of the movie it was awarded to.
8. Which actor(s) were also directors to one or more movies? Return their person ID (PID) and last names.
9. A valid xml (satisfying the DTD in file **stats.dtd**) that contains a histogram showing the number of movies that appeared in each genre (main category).
10. A valid xml (satisfying the DTD in file **actressinfo.dtd**) that contains information about the actress who appears in more movies than any other actress. If there is a tie, report all actresses that tied.

Store each query in a separate file, and call these **q1.xq** through **q10.xq**.

For the queries that generate XMl output, generate only the XMl elements. Don't try to prepend that with the "header" information that belongs at the top of the file.

We will not be autotesting your code, so some of these questions above are not as specific about format as they would have to be if we were autotesting. The exceptions are queries where you are to produce an XML file that conforms to a DTD. These DTDs will be posted on the Assignments page. In all cases, white space is up to you. Just try to make your output readable.

Here are a few XQuery tips that may help:

- Although we spent most of our time on FLWOR expressions, there are other kinds of expression. We've studied `if` expressions, `some` expressions and `every` expressions. And remember that a path expression is an expression in XQuery also.
- XQuery is an expression language. Each query is an expression, and we can nest expressions arbitrarily.
- You can put more than one expression in the `return` of a FLWOR expression if you put commas between them and enclose them in round brackets.

- XQuery is very "fiddley". It's easy to write a query that is very short, yet full of errors. And the errors can be difficult to find. There is no debugger, and the syntax errors you'll get are not as helpful as you might wish. A good way to tackle these queries is to start incredibly small and build up your final answer in increments, testing each version along the way. For example, if you need to do the equivalent of a "join" between the data files, you could start by iterating through just one of them; then make a nested loop that makes all pairs; then add on a condition that keeps only the sensible pairs. Save each version as you go.

3 Capture your results

Capture your results by running the script `runall.sh` found on the course website. It will run galax-run on each of your queries and sends the output to a file called results.txt.

When doing your own testing, you may make a modified version of the script if you find that helpful. But make sure you hand in results from the original script. If you are not able to complete a query, you may comment it out. Because of this possibility, I will ask you to hand in the script as well.

4 Electronic Submission Instructions

You should submit the following files electronically using the submit command in CDF:

- Information about your team (whether it is a team of one or two students): `team.txt`
- Your query files: `q1.xq` through `q10.xq`
- Your "capture" script: `runall.sh` (even if you didn't change it)
- Your results file: `results.txt`

You may work at home, but you must make sure that your code runs on the CDF machines.

When you have completed the assignment, move or copy your files in a directory (e.g., assignment4), and use the following command to electronically submit your files within that directory:

% submit -c csc343h -a A4 team.txt q1.xq q2.xq q3.xq q4.xq q5.xq q6.xq q7.xq q8.xq q9.xq q10.xq runall.sh results.txt

You can also submit the files individually after you complete each part of the assignment - simply execute the submit command and give the filename that you wish to submit. You may submit your solutions as many times as you wish prior to the submission deadline (you might need to use the -f flag of the submit command). Make sure you name your files exactly as stated (including lower/upper case letters). Failure to do so will result in a mark of 0 being assigned.

Once you have submitted, be sure to check that you have submitted the correct version of each file; new or missing files will not be accepted after the due date. You may check the status of your submission using the command

% submit -l -N A4 csc343h

where -l is a hyphen followed by the letter 'ell'.