Programming Project 4
Due: Friday, April 29, 11.59PM (Blackboard)

Rigel Gjomemo

April 23, 2016

1 Description

In this part of the project, you will test the database that you built for your bookstore application with a very large volume of data. In order to avoid the web communications overhead, you will use a simple java program (written in a simple Java file) to get in input the number of tuples in the tables and to populate those tables. This java file, together with a text file with the create database and create table statements, as well as a report must be submitted on Blackboard.

2 PART 1 (40%) Create a large database

In this part you must write JAVA code to populate the bookstore database tables with an arbitrary number of tuples specified from the command line.

At the start of the program, the user must be asked the number of tuples that will be created for each table. In other words, the program must present several questions to the user and read the corresponding number of tuples from the user, for every table. The number of questions depends on the design of your database and on the number of tables in your database. At a minimum, you must create tuples about books (book title, price, ISBN, etc), authors (name, id, etc), website users (username, address, etc), books written by authors, and order history. For example, in my bookstore application I have the tables `users, authors, books, booksAndAuthors, order_history`. Therefore, running the program would look something like this:

```
How many tuples would you like in the authors table:  1000
1000 tuples created in the books table.

How many tuples would you like in the order_history table:  200000
200000 tuples created in the order_history table.

How many tuples would you like in the books table:  10000
10000 tuples created in the books table.
```

```
How many tuples would you like in the users table:  20000
20000 tuples created in the users table.

How many tuples would you like in the booksAndAuthors table:  10000
10000 tuples created in the booksAndAuthors table.
```

Create random values for the tuples. For instance, populate the users database with tuples of the type (`user_id1, username1, address1, ...`), (`user_id2, username2, address2, ...`), (`user_id3, username3, address3, ...`), and so on.

Remember that there exist some foreign key constraints on the tables, so you must make sure in your program that they are not violated. For instance, when inserting in `order_history`, you must make sure that the value of the `ISBN` exists in the tables `books`, and so on. I suggest that you choose a simple randomization scheme for inserting the values of those columns that are referred by other columns, so that you know the range of possible values. For instance, if you know that all the `user_ids` fall in the range `user_id1 - user_id10000`, you can generate a random number between 1 and 10000 and append it to the string `user_id` when inserting a tuple in the `order_history` table. The alternative would be to query the database for existing values but that would double the number of queries.

In addition, add a year of birth column to both the users and authors table. This should be an integer between 0 and 4000 for the authors, and between 0 and 6000 for the users. Add also a column in the books table to store the year in which the book was written. This should be a number between 0 and 4000. As you may have guessed, this is a database about a world in which there are time machines for everybody. If you choose these values randomly, you will end up with books written before their authors were born and bought by users before they were written, and other such combinations but do not be bothered by this.

3 Part 2 (60%)

Using the program from part 1, create three databases of increasing size. For all the databases, the book prices must be randomly chosen between $1 and $100 and the order times must be randomly chosen between midnight of the year 0 and midnight of the year 6000.

1. Bookstore1
    - 100 authors
    - 1000 books
    - 100 users
    - 2500 orders
2. Bookstore2
    - 1000 authors
    - 10000 books

- 1000 users
- 25000 orders

3. Bookstore1
    - 5000 authors
    - 50000 books
    - 10000 users
    - 100000 orders

Run the following queries on your databases:

Q1. Find all the usernames of those users that bought books in the year 2005, and where those books were written in the years 2016, 3333, 1200 from authors born in 1999 and where the books have a price greater than 50 dollars.

Q2. Find the orders made between the years 0 and 3000 by users born between the years 15 and 1100 for books written after the year 2000 and having a price of $15.

For every query, record the time it takes for the query to execute on every database.

Next, modify the tables by adding indices to the appropriate columns, in order to improve the execution times. Try both with B-Tree indices and with hash indices. Report the execution times before and after the modifications. Describe in your report the output of the MySQL 'explain' command on each of your queries (`http://dev.mysql.com/doc/refman/5.7/en/explain.html`).

4 Deliverables

1. **Code**. Your code (the Java file that reads the user input and creates the databases, and the text file with the create database and create tables statements), and your report should be submitted on Blackboard by the deadline.