School of Computer Science, McGill University

COMP-421 Database Systems, Winter 2017

Project 3: Writing your Application

Due Date March 23, 11:59pm

This is the last deliverable for your database application project.
The TA responsible for the Project 3 deliverable will be Faizy.

1. (15 Pts) Write one stored procedure to perform operations on your project database. It should be nontrivial, illustrating a feature or features such as local variables, multiple SQL statements, loops etc. It should also involve a cursor. The stored procedure should use one or more parameters in a significant way. We encourage you to be imaginative. However, here are some sorts of things you might try if you cant think of something more interesting:

   - Compute some aggregate value from a relation and use that value to modify values in that or another relation.
   - Create a new relation and load it with values computed from one or more existing relations.
   - Enforce a constraint by searching your database for violations and fixing them in some way.

   Hand in a listing of your programs and scripts showing them working. You should demonstrate that the programs had their intended effect by querying (before and after) some relation of your project database that was changed by the program. These queries may be included in the file that holds your programs for convenience.

2. (50 Pts) Write a user-friendly application program for your database in Java (You may use another language like python as long as you can sort out the connectivity to the database). There is no need for a fancy interface. For example, a menu printed via simple I/O is ok.
   Your program should consist of a loop in which:

   - A list of at least five alternative options is offered to the user. An additional alternative should be quit.
   - The user selects an alternative.
   - The system prompts the user for appropriate input values.
   - The system accesses the database to perform the appropriate queries and/or modifications.
   - Data or an appropriate acknowledgment is returned to the user.

   Your program should follow the following guidelines.

   - Your options should include both queries and modifications.
   - Some of your options should contain more than one SQL statement.
   - Your program must handle errors appropriately. For Java, catch exceptions and print the error messages.

   For example, if your project were about skaters and competitions.

   - Look up whether a skater participates in a certain competition by skater name.
   - Enroll a skater S in a competition C. If the rating level is below 3, S cannot enroll in any competition. If it is between 3 and 6, S can enroll in regional competitions only, if its is between 7 and 9, he/she can enroll in regional and national levels, and only with a skating level of 10 can S enroll in all types of competitions. If S is not qualified for the competition C, return a list of alternative competitions for which the S has the minimum rating level and which are close to C in terms of the date.

- A competition is cancelled: find all skaters participating and replace the participation with a competition close in time to the cancelled competition.
- Add a new skater.
- Increase the rating of skaters that were among the first 5 in at least 2 competitions of the highest level they can participate.
- ...
- Quit

Hand in your program and a script showing the program running. Each of the options should be exercised at least once in your script.
Make sure to include screen shots of the program executing various options(similar to Project 2 deliverable).
You may skip the screen shots in your project report for this question if you had instead decided to do a demo to the TA.

3. (10 pts.) In class we discuss indexes that help to speed up queries. You can create and drop an index using: commands:

```
CREATE INDEX <IndexName> ON <RelName>(<Attribute List>);
e.g., CREATE INDEX skatersname ON Skaters(sname);
DROP INDEX <IndexName>;
```

Statements for more sophisticated indexes (unique, clustered etc.) can be found in the lecture notes and also in the DBS manuals.

Create at least two useful indexes for your project database. Do not build indexes on primary keys and unique keys. Database systems usually create indexes on these attributes automatically as they need them to check the uniqueness property. For each of the created indexes indicate why this index is useful by describing which application relevant queries would execute quicker

4. (15 pts.) In class we discussed how database uses some statistical information ... in this part we will do some simple exploration.

First create a copy of one of your larger (number of tuples) project tables. We will work on this copy for this part of the assignment.

After that,

Make the database collect statistics on one of the columns in the table (explore the ANALYZE command in postgresql and RUNSTATS command in DB2)

Now let us see what the database expects the cardinality of the table will be. To do this, in postgresql do a simple `EXPLAIN SELECT * FROM mytable` Check to see what the database thinks is the cardinality of the table is.

In DB2, you have to execute this command from the shell prompt.
`db2expln -database cs421 -t -g -q "SELECT * FROM mytable"` Verify the cardinality estimates of the database (ignore other information provided).

Document these results.

Now delete or add some records to your table.
DO NOT run the statistics commands yet, but check what the database thinks the cardinality is now. Did it change ? Document the results and your remarks.

Now run the statistics commands again. Verify the cardinality estimates of the database. Document the results and your remarks.

Similar to the other deliverables, turn in the commands you executed, their outputs as well as a summary of your observation and conclusions.

5. (10 pts.) For creativity points, you may explore any one of these (some are topics not covered in class).

   - Triggers - create a trigger that does something interesting for your application logic - give a brief description of its purpose.
   - A sophisticated GUI (instead of the console based UI for question 2, but does the same operations)

- An extra stored procedure (it should do something totally different from the one required in this deliverable)

If something else interests you for creativity points, come talk to me.

If you are doing a GUI for creativity points, it is highly recommended to sign up for a demo slot. As stated before, if you are doing a demo for question 2, then you do not have to include the screen shots of the program execution for question 2. You still need to turn in your programs/scripts.
You should still turn in your project report by the due date.

Note : The project design was taken and adjusted from a project description of the CS145 Stanford database course.