ECE 356 Assignment 1

Basic Admin, Client-side SQL, Relational Algebra

Due: September 28th, 2021

The purpose of assignment is to learn how to

- use the MySQL Command-Line Interface (CLI)
- create basic SQL queries involving a single table
- write a client program in C that can access a database
- create Relational Algebra queries to answer natural-language questions

1 The Command-Line Interface

Database engines vary in small but significant ways; in this course we will endeavor to keep most things engine-agnostic. One of the features that is common across relational databases is the command-line interface (CLI) as it is a defined part of the standard. CLI is not just of value because it is a shell available for every relational database, but also because it is very useful for database automation.

Sort of; it is in the standard, but the degree to which implementations conform to the standard varies and so the specific CLI is not always identical across systems. MySQL is arguably one of the easier ones to learn. Postgres is one of the uglier ones.

The following instructions assume that you do not have your own MySQL instance or MySQL client instance, and will use university equipment. Also, some aspects of these instructions are recommended, but optional; if you plan to do them, it is strongly recommended that you start by doing that first, and then continuting with this material.

You will probably also want to set up SSH keys if you have not done so in a prior course, unless you enjoy enetering your password a lot.

To access the CLI using campus equipment you will need to log into either `eceubuntu` or `ecetesla`. To do so you will need to be connected to the campus VPN. If you have never connected to the campus VPN before, please read the material at

`https://uwaterloo.ca/information-systems-technology/services/virtual-private-network-vpn`

and ask questions on Piazza. Once you have logged in to `eceubuntu` or `ecetesla` you should then issue the command:

```
mysql -u <userid> -p -h marmoset02.shoshin.uwaterloo.ca
```

where `userid` is the userid you have been given by e-mail. You will be prompted for a password. The password you enter is the MySQL password that was included in the e-mail and it is highly recommended that the first thing you do is change your password using the command:

It will be your University of Waterloo 8-character UserID, but the password is specific to the MySQL server as that server is not connected to the UW ADFS credentials service.

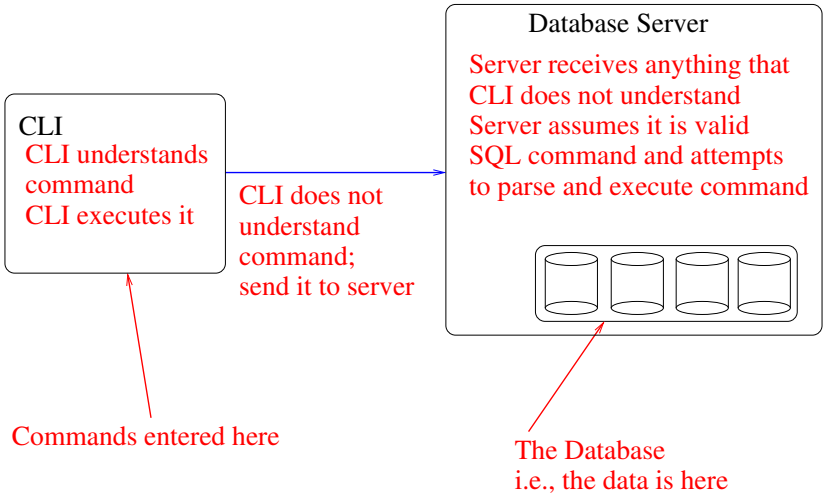
```
set password = '...';
```

Detailed instructions on changing your password can be found at:

`https://dev.mysql.com/doc/refman/8.0/en/set-password.html`

Detailed information about CLI commands can be found at:

`https://dev.mysql.com/doc/refman/8.0/en/mysql-commands.html`



The operation of CLI is as illustrated above. Briefly, there are a number of commands that it understands, such as `use`, `status`, *etc*. However, it does *not* understand SQL. If it receives a command that it does not understand it assumes that it is an SQL statement and will send it to the database server. On receiving anything from the CLI, the server will attempt to parse it as SQL and, if it is valid SQL, will then execute the command.

CLI commands follow the standard "scripting language" convention using a newline to mark the end of a command. SQL, by contrast, uses the standard C convention and uses a semicolon (";") to mark the end of an SQL statement. If you enter an SQL statement into the CLI and do not get any response, chances are you forgot the semicolon. I know I forget it *all the time*. Ugh.

Some commands do not look like SQL statements and so you might think they are CLI commands and will be executed in the client and not sent to the database server. For example, the statement:

```
show databases
```

is not valid SQL. These statements are MySQL "syntactic sugar" for more complex SQL statements. In the particular case of "`show databases`", it is a MySQL convenience that represents the SQL query:

```
select schema_name as 'Database' from information_schema.schemata order by schema_name;
```

You can use the "`help`" command to determine what commands CLI understands and "`help contents`" to determine the commands that the database server understands. Detailed information about server-side CLI commands can be found at:

`https://dev.mysql.com/doc/refman/8.0/en/mysql-server-side-help.html`

Task 1: CLI and SQL Commands

For this task, after starting CLI, you must execute the command:

```
tee cliSession.txt
```

This will create and open a file called `cliSession.txt` and cause CLI to record your interactive session with it. When you have completed the CLI and SQL commands you should execute the command:

If a file of this name already exists, you should rename it or delete it, at your discretion, as the "`tee`" command will append to a file if it already exists.

```
notee
```

which closes the file. This file forms part of your solution submission.

It is a good discipline to do this whenever you are doing interactive CLI sessions as you will then have a recording of your interaction so that you do not lose track of what you did. It is, essentially, the equivalent of an experimental scientist's logbook.

You should now determine and execute CLI and/or SQL commands so as to be able to answer the following questions. In each instance you should identify whether the command is a CLI command executed on the client-side or an SQL command executed at the database server. Your answers are to be recorded in a CSV file called "`task1.csv`" with the following format:

```
questionNumber, "Command", CLIorSQLcommandType, "Answer"
```

For example, if you executed the hypothetical CLI command "`foo foo`" in order to answer the first question, and the answer you discovered was "`foo bar`" then the first line of your "`task1.csv`" file would be:

```
1, "foo foo", CLI, "foo bar"
```

These are strict requirements and deviation from them requires manual grading of your solutions. As such, there will be a penalty if you fail to follow them.

1. What is the version number of the database server?
2. How many databases do you have access to?
3. What are the names of the databases that you have access to?
4. The following questions pertain to the "Loyal" database:
   (a) How can you connect to the "Loyal" database?
   (b) What is the other way that you can connect to the "Loyal" database?
   (c) How many tables are in the "Loyal" database?
   (d) Which table has the fewest attributes?
   (e) What are the types of the table with the fewest attributes?
   (f) How many products cost between $150 and $450?

   For these questions, the question number should be identified as 4a, 4b, *etc.* in your `task1.csv` answer set.
5. Exit the CLI and then restart it. After restarting, the first command you must execute is
   `tee cliSession.txt`
   so that this second CLI session will append its interactive output to your `cliSession.txt` file.
   What database are you connected to?
6. Determine how many products cost between $150 and $450 in the "Loyal" database *without* first connecting to that database.

2 *Task 2: Embedding SQL in a client program*

A sample C/SQL program that connects to the database server and implements a subset of the "`show databases`" command will be provided for you on Learn. This command can restrict its output with a "`like`" subclause, thus:

```
show databases like '<pattern>';
```

This performas a pattern match on the database name, where the percent symbol ('%') is a wildcard match. Only databases with matching names will be

listed. There is, however, no version of "`show databases`" that allows negative matching. That is, there is no command that does:

```
show databases not like '<pattern>';
```

You are required to take the sample program and modify it so that it implements such a negative pattern match. Call your program "`task2.c`".

3 Task 3: Relational Algebra Queries

Throughout this course we will be using a NHL Hockey database as a running example and for assignment questions. This database originated at NHL.com and a subset of the NHL database relational schema will be provided on Learn and Piazza in a file called "`NHLschema.pdf`". While you are not expected to become an NHL expert, you are expected to become somewhat familiar with the game. This requirement is consistent with typical database practice where the database expert is not expected to be the *domain* expert but will be expected to become somewhat familiar with the domain. Domain-specific questions should be posed on Piazza and the need to find answers to these questions *is* considered part of the assignment. It is *NOT*, however, something for which you should worry about plagiarism offenses. For the purposes of this aspect of the assignment, we view the entire class as a single team, collectively attempting to learn enough about the domain so as to understand enough of the particular database to determine appropriate Relational Algebra queries.

`https://statsapi.web.nhl.com/` specifically

In this context, *you* are the database expert.

You are required to create (extended) relational-algebra queries to answer the following questions:

1. When was Sidney Crosby Born?
2. How many teams are in the NHL?
3. All NHL teams are based in either Canada or the United States. However, NHL players come from (were born in) many different countries around the world. How many different nations are there represented in the NHL?
4. What fraction of players were born in Canada?
5. What fraction of players were born in December?
6. How many players who played games in the 2008/2009 season also played games in the 2018/2019 season?
7. What is the greatest number of penalties given in a game, and who were the teams playing in that game?
8. What is the greatest number of penalties given in a game in the 2018/2019 season, and who were the teams playing in that game?
9. What player, who played in both the 2008/2009 season and also played in the 2018/2019 season, received the greated number of penalty minutes in the 2018/2019 season?
10. How many players who played in both the 2008/2009 season and the 2018/2019 season had more penalty minutes in the 2018/2019 season than they had in the 2008/2009 season?

4 Task 4: Meta-Cognition

In simple English: students with an accurate self-assessment learn material better than those with poor self-assessment.

There is evidence in the learning research literature that it is not only important to be able to solve problems, but it is also important to be able to know how well you have performed. For this task, you are required to create a file, "`task4.csv`", that contains three lines of form:

```
taskNumber, estimatedPerformance
```

where the estimated performance is how well you think you did, as a percentage, on the particular task. For example, there are 11 questions in task 1 and so if you think you got 8 of the 11 questions basically correct, but think you did badly on three of them, you would record that line as:

```
1,73
```

The evaluation of this meta-cognition task is based on how accurately you assess you expected performance.

5 Deliverables

A DropBox will be opened on Learn for you to submit the following items:

1. cliSession.txt containing the `tee` output of your CLI session.
2. task1.csv containing your answers to the CLI/SQL questions for task 1.
3. task2.c containing your C/SQL code
4. task3.pdf containing your written answers to the RA questions above.
5. task4.csv containing you meta-congition self-assessment.
6. (optional) README.md containing any information you choose to submit for the assignment grader to consider when marking your assignment.

6 Material on Which This Assignment is Based

This assignment is based on materials covered in Modules One and Two. In addition, you will need to consult the MySQL reference manual.