DUE DATE: **MONDAY/TUESDAY, March 30/31, 2020 during Lab time**

Introduction to Systems Programming (System I)

Lab #6

Max Points: 50

You should save/rename this document using the naming convention MUid.docx (example: ahmede.`docx`).

Objective: The objective of this exercise is to gain experience with running SQL commands at the `mysql` command-prompt:

1. Connecting to `mysql` from a terminal
2. Creating a table in a database.
3. Inserting records into a table.
4. Querying for data in tables.

Fill in answers to all of the questions. For some of the questions you can simply copy-paste appropriate text from the terminal/output window into this document. You may discuss the questions with your instructor.

Name: **Ben Hilger**

Wait for your instructor to indicate when you can start working on each part of this exercise. Your instructor will cover necessary concepts prior to working on this laboratory exercise. Your instructor may have you work on one question at a time before moving on to the next question.

Part #1: Connecting to MySQL database on `os1.csi`

Estimated time: 12 minutes

Background: MySQL is a network-enabled RDBMS that accepts and processes commands over a network socket. MySQL typically listens on port 3306. MySQL uses a custom protocol for sending & receiving data. Since the protocol is binary, it is cumbersome to use `telnet`. Consequently, special command-line tool called `mysql` (don't confuse between `mysql` a software-tool and MySQL the RBDMS which is a collection of software tools) is typically used to interactively work with MySQL. Later on we will study using C++ library to interface with MySQL.

Exercise: Connecting to MySQL RDBMS on `os1.csi.miamiOH.edu`

1. From a Powershell/Terminal use `ssh` to log onto `os1.csi.miamioh.edu`

2. On `os1.csi` connect to MySQL RDBMS using the following command (**all on 1 line**):

```
mysql --protocol=TCP --user=cse278s19 --password=rbHkqL64VpcJ2ezj --
database=cse278s19
```

   Notes:
   - `--user`: Indicates MySQL user ID, which is typically different than your login ID for security reasons. Correct, in most companies their data & databases are more valuable than your login ID. Typically administrators will have different user ID.
   - `--password`: MySQL password associated with the user ID. Typically you will never expose the password like this.
   - `--database`: The name of the database to use by default. In this case both the use ID and database happen to have the same name. This is just for convenience and not a requirement.

3. Upon successfully connection to MySQL RDBMS you should see some start-up messages and you will be provided a "`mysql>`" prompt to type SQL commands. Most RDBMS work this way. We are using a simple command-line interface. There are software products you can buy that will provide a GUI to access and work with the database. In this course will we will work as professionals and use the raw textual interface -- *i.e.*, the professional "all stuff, no fluff" interface.

4. To log out of `mysql`, use the `exit` command as shown below:

```
mysql> exit ↲
```

☞ NOTE: You will need to complete this part of the exercises successfully prior to proceeding with remainder of this exercise. So if you get error messages ensure you get help from your instructor to resolve the error.

Part #2: Creating a table

Estimated time: 10 minutes

Background: Prior to storing data in a RDBMS, a table with suitable schema must be created.

Exercise: Create a table with your MUid (e.g., ahmede) with the following procedure

1. If needed, log onto MySQL RDBMS using the `mysql` command-line introduced earlier.
2. NOTE: Everywhere you see `MUid` you should use your login id instead.
3. Create a table to store person information (namely: id, name, and email) by appropriately changing `MUid` in the following command (correctly copy-pasting will work fine):

```
CREATE TABLE hilgerbj (
   id    INTEGER NOT NULL,
   name  VARCHAR(64) NOT NULL,
   email VARCHAR(32) NOT NULL,
   PRIMARY KEY (id)
);
```

4. If the table is successfully created, you should get the message "`Query OK, 0 rows affected`". If you get an error, get help from your instructor to resolve the error.

☞ NOTE: You will need to complete this part of the exercises successfully prior to proceeding with remainder of this exercise. So if you get error messages ensure you get help from your instructor to resolve the error.

Part #3: Insert some data into your table

Estimated time: 10 minutes

Background: One of the key operations of working with a database is inserting new values into the database. This is accomplished using SQL `insert` statement.

Exercise: Insert sample data into your MUid (NOTE: Everywhere you see `MUid` you should use your login id instead) table that you created in the previous step with the following command:

1. If needed, log onto MySQL RDBMS using the `mysql` command-line introduced earlier.

2. Insert a row into the table using the SQL `insert` statement below:

```
INSERT INTO hilgerbj (id, name, email) VALUES (100, 'John
Doe', 'jdoe@unknown.com');
```

3. Check to ensure that you get "`Query OK, 1 row affected`" as the response indicating 1 row was successfully inserted. If you get errors get help from your instructor to resolve the issue.

4. Suitably modify the insert command to insert the following 3 rows to your table:

id	**name**	**Email**
200	'Mary Doe'	mdoe@unknown.com
300	'James Bond'	jb@mi6.gov.uk
400	'Superman'	sman@miamioh.edu

NOTE: You will need to complete this part of the exercises successfully prior to proceeding with remainder of this exercise. So if you get error messages ensure you get help from your instructor to resolve the error.

Part #4: Experiment with SQL queries

Estimated time: 10 minutes

Background: Querying data using SQL involves coding suitable SELECT statements with necessary columns and conditions.

Exercise: Using the SELECT statements discussed in class, code the SQL statement for the following queries:

1. Develop a SQL query to list all rows in your table (copy-paste your SQL statement and output in the space below). This query is already implemented for you to illustrate an example.

```
mysql> select * from MUid;
+-----+------------+------------------+
| id  | name       | email            |
+-----+------------+------------------+
| 100 | John Doe   | jdoe@unknown.com |
| 200 | Mary Doe   | mdoe@unknown.com |
| 300 | James Bond | jb@mi6.gov.uk    |
| 400 | Superman   | sman@miamioh.edu |
+-----+------------+------------------+
4 rows in set (0.00 sec)
```

2. Develop a SQL query to list just `name` of persons with ID less than 400 (copy-paste your SQL statement and output in the space below):

```
SELECT name from hilgerbj where id < 400;
```

3. Develop a SQL query to list `id` and `name` of persons that have the word `'Doe'` in their name using the `LIKE` comparison operator (copy-paste your SQL statement and output in the space below):

```
SELECT id, name from hilgerbj where name LIKE '%Doe%';
```

4. Develop a SQL query to list `id` and `name` of persons, with data sorted by `name` (copy-paste your SQL statement and output in the space below):

```
SELECT id, name from hilgerbj ORDER BY name;
```

Part #4: Data normalization

Estimated time: 10 minutes

Exercise:
In the space below, briefly (3-to-4 sentences) describe what is "Data normalization" and why is it important?

```
Data normalization is reorganizing data in a database into a coherent and efficient
way to allow for easy queries and analysis. Data normalization is important because
it provides a much more organized and consistent and reduces redundancy. It also
improves data integrity since it keeps information in one spot, allowing
information to be updated and referenced much more easily.
```

What is the difference between *normalized* and *denormalized* databases?

```
The difference between normalized and denormalized databases is that denormalized
databases focus on improving the read performance of databases at the loss or write
performance and overall organization. Whereas normalized databases focus on
organization and improving write performance at a loss to read performance.
```

Part #5: Experiment with SQL queries on 2 tables

Estimated time: 15 minutes

Background: Normalized databases will involve data that is organized into 2 or more tables. However, to answer some (not all) queries, the data from 2 (or more) tables may need to be *joined* together (using PK and FK) to get the relevant information. Join operations in SQL are accomplished in the following manner:

1. Indicate the tables involved in the query in the FROM clause
2. Specify relationship/condition between columns in from the tables.

Exercise: For this part of the exercise the following tables have already been created and populated with data for you:

```
Product (pname,  price, category, manufacturer)
Company (cname, stockPrice, country)
```

1. Develop a SQL query (no join needed) to list all rows & columns in `Company` table (copy-paste your SQL statement and output in the space below).

```
SELECT * from Company;
```

2. Develop a SQL query (no join needed) to list all rows & columns in `Product` table (copy-paste your SQL statement and output in the space below).

```
SELECT * from Product;
```

3. Develop a SQL query to list all names of products (`pname`) manufactured by Japanese companies (copy-paste your SQL statement and output in the space below).

```
SELECT pname from Product, Company where cname=manufacturer AND
country="Japan"
```

4. Develop a SQL query to list names and stock prices of companies that manufacture products cheaper than $200 (copy-paste your SQL statement and output in the space below).

```
SELECT DISTINCT cname, stockprice from Product, Company where
cname=manufacturer AND price < 200;
```

Submit to Canvas

- No late assignments will be accepted!
- This work is to be done individually
- This MS-Word document (duly filled-in) saved as a PDF document.
- The submission file will be saved with the name ***Lab6_yourMUID.pdf***
- Assignment is due Monday/Tuesday, March 30/31 during Lab time
- On or before the due time, drop the *electronic copy* of your work in the *canvas*

Don't forget to Turn in the files!   Lab6_yourMUID.pdf