

## Lab 8 Mini Labs

**Due date:** June 2<sup>nd</sup>.

### Assignment Preparation

This is an individual lab. Each student has to complete all work required in the lab, and submit all required materials **exactly as specified** in this assignment.

The assignment will involve writing small commands/programs to test basic knowledge of a variety of database-related topics.

### The Task

#### Database Connectors

Your task is to use a database connector to interact with our class database. You can use whatever language you choose, but there are some restrictions:

1. Your program must be able to be run on the csl machines.
2. Your program must connect to the class database (csc-db0) with your standard credentials.
3. You must have a script called `run` (with no extension) that will compile (if necessary) and run your program. Assume no environmental variables are set (like `CLASSPATH` which is important for java). You may have to bundle libraries (like the MySQL connector jar for Java users) with your submission.

Put all your files in a subdirectory called **connector**.

#### Specifications

You will be writing a program that interacts with the STUDENTS dataset and answer questions similar to the ones you answered in Lab 1.

Specifics:

- Your program should take no command-line arguments.
- It should have **NO PROMPT**. The only output will be the results of the command.
- You do not have to worry about injection or malformed input for this lab.
- All commands are case sensitive.

- Do not have a space between the command and the colon.

You will have to implement the following queries:

- **S[tudent]: <first name>**

When this instruction is issued, your program shall perform the following:

- Find all the students with the given **first** name.
- For each student, print:  
     <first name>, <last name>, <grade>, <classroom>, <teacher first name>, <teacher last name>

- **T[eacher]: <last name>**

When this instruction is issued, your program shall perform the following:

- Find all the teachers with the given **last** name.
- For each teacher, print:  
     <first name>, <last name>, <classroom>, <number of students they teach>

- **C[classroom]: <number>**

When this instruction is issued, your program shall perform the following:

- Find the classroom with the given number.
- For each classroom, print:  
     <classroom>, <teacher first name>, <teacher last name>, <number of students in the classroom>

- **Q[uit]**

Just quit. Perform no additional prompts.

Example output (user input is **bold**):

```

S: GAYLE
GAYLE, GAYLE, 4, 111, BILLIE, KRIENER
GAYLE, GAYLE, 4, 110, GEORGETTA, SUMPTION
T: COVIN
JEROME, COVIN, 102, 7
CLASSROOM: 102
102, JEROME, COVIN, 7
Q

```

## Security

For this section, you will be doing various security-related activities. For the first two parts of this section, put your SQL statements in a file called **security.sql**.

## Transactions

Write a **fully serialized** transaction to model the corporate takeover by Northwest of all flights except those operated by Virgin and Jet Blue. See the **AIRLINES** dataset in Lab 3. (You do not need to worry about deleting the flights not in/out of AOS).

## GRANT

Write the following GRANT statements (if not specified, assume that the user is accessing the database from any machine):

1. Give SELECT privileges to the account **selectUser** (coming from the local machine) on all tables in the **Dinosaurs** database.
2. Give the **newRoot** user root level privileges on the database server. (You will have to work out what privilege a root user should have).
3. Give the user **dan** privileges to do anything he pleases in the **DansDatabase** database.
4. Give any user coming from the local machine root level privileges.
5. Give any user coming from a machine on the local network the ability to alter any tables, as long as the provide the password 'password'.

## SQL Injection

Attck the following URL:

<http://users.csc.calpoly.edu/~eaugusti/pages/365-spring16/injection.php?attrs=firstName,lastName,email&user=eriq>

Your goal is to find out the **usernames** and **passwords** for all the users in the system. Note: first and last name are **NOT** usernames. When you find the requested data, put it in a file called **users.txt** one user to a line, in the following format:

**<username>,<password>**

You will be given no additional hints or specifications.

## Stored Procedures

For this section, you will have to write two stored procedures/functions and put them both in a file called **procedures.sql**.

**medianItems** Write a function called **medianItems** that finds the median number of items purchased over each BAKERY transaction. This function takes no parameters, and must return the median as a FLOAT.

**goodsStats** Write a procedure that finds the min, max, mean, and median price for all the goods in the BAKERY dataset. This procedure must return all the requested values as **out parameters** in the following order: min, max, mean, and median.

Note for all median calculations: In the event of an even number of values, take the mean of the two middle values.

## Submission Instructions

**Please, follow these instructions exactly.** Up to 20% of the Lab 8 grade will be assigned for conformance to the assignment specifications, **including the submission instructions.**

Please, **name your files exactly as requested** (including capitalization). Correct submission simplifies grading, and ensures its correctness.

### Specific Instructions

You must submit all your files in a single archive. Accepted formats are **tarball (gzipped tar)** (**.tar.gz**) or **zip (.zip)**. The file you are submitting must be named **lab08.ext**, i.e., either **lab08.tar.gz** or **lab08.zip**

Inside it, the archive shall contain a directory called **connector** and the files **procedures.sql**, **security.sql**, **users.txt**, and **README.txt**. The **connector** folder must contain the **run** script, and all other resources you need for that portion of the lab.

It is **INCORRECT** for a submission to unpack into a single directory which then contains the actual submission.

### Example Directory Structure

Your archive should unpack into the same structure (except for the specific contents of the **connector** directory ) outlined below:

```
.
|-- README.txt
|-- connector
|   |-- connector.rb
|   '--- run
|-- procedures.sql
|-- security.sql
'-- users.txt
```

### Handin

Once you created your submission archive, submit it using the following **handin** command:  
\$ handin eaugusti 365-lab08 <ARCHIVE>