CNT 4714 – Project Three – Fall 2021

Title: "Project Three: Two-Tier Client-Server Application Development With MySQL and JDBC"
Points: 100 points
Due Date: Sunday October 24, 2021 by 11:59 pm (WebCourses Time)

Objectives: To develop a two-tier Java based client-server application interacting with a MySQL database utilizing JDBC for the connectivity. This project is designed to give you some experience using the various features of JDBC and its interaction with a MySQL DB Server environment.

Description: In this assignment you will develop a Java-based GUI front-end (client-side) application that will connect to your MySQL server via JDBC.

You are to develop a Java application that will allow any client (the end-user) to execute commands against the database. You will create a Java GUI-based application front-end that will accept any MySQL DDL or DML command, pass this through a JDBC connection to the MySQL database server, execute the statement and return the results to the client. Note that while technically your application must be able to handle any DDL or DML command, we won't actually use all of the commands available in these sublanguages. For one thing, it would be quite rare to allow a client to create a database or a table within a database. Note too, that the only DML command that uses the `executeQuery()` method of JDBC is the Select command, all other DML and DDL commands utilize `executeUpdate()`. Some screen shots of what your Java GUI front-end should look like are shown below. Basically, this GUI is an extension of the GUI that was developed in the lecture notes and is available on WebCourses as DisplayQueryResults.java. Your Java application must give the user the ability to execute any SQL DDL or DML command for which the user has the correct permissions. Note also, that if the user wishes to change databases in the middle of a session, they must reconnect to the new database. Their user information can remain in the proper window, but you must click the reconnect button to establish a connection to the new database. You will be able to start multiple instances of your Java application and allow different clients to connect simultaneously to the MySQL DB sever, since the default number of connections is set at 151 (See your Workbench options file under the networking tab). In addition to the client interactions with your application, a background (business logic) transaction logging operation will occur which keeps a running total of the number of queries and the number of updates that have occurred via the user application. This is a separate database (i.e., a completely different database than any tto which the client user can connect), that the application will connect to using root user privileges. Each user operation will cause the application to make this connection and update the operational logging database table. More details on this aspect of the application are shown below and will be covered in the Q&A sessions.

Once you've created your application, you will execute a sequence of DML and DDL commands and illustrate the output from each in your GUI for two different users. For this project you will create, in addition to the root user, a client user with limited permissions on the database (see below). The root user is assumed to have all permissions on the database, any command they issue will be executed. The client user will be far more restricted.

References for this assignment:

Notes: Lecture Notes for MySQL and JDBC.

Input Specification:
The **first step** in this assignment is to login to the MySQL Workbench as the root user and execute/run the script to create and populate the backend database. This script is available on the assignment page and is named "`project3dbscript.sql`". This script creates a database named **project3**. You can use the MySQL Workbench for this step, or the command line whichever you prefer. This script file is available on WebCourses.

The **second step** is to create authorizations for a client user (in addition to the root user) named `client`. By default your root user has all permissions on the **project3** database. Use either SQL Grant statements from the command line or the MySQL Workbench (see separate document for details on how to accomplish this task) to check and set permissions for the client as follows:

Register the new user named **client** (assign them the password ***client*** – ignore the MySQL warning on weak password setting) and assign to this user only selection privileges on the **project3** schema.

The **third step** is to create the **operationslog** database using the `project3operationslog.sql` script. This script file is also available on WebCourses.

Output Specification:
There are three parts for the output for this project. Part 1 is to provide screen shots from your application which clearly show the complete query/command expression and results for each of the commands that appear in the script named: **`project3rootuserscript.sql`** available on the course website. There are eight different commands in this script and some of the commands will have more than one output capture (see below). Part 2 is to provide screen shots from your application which clearly show the complete query/command expression and results for each of the commands that appear in the script named: **`project3clientuserscript.sql`** available on the course website. There are three different commands in this script and some of the commands will have more than one output capture (see below). **To produce your final output, first recreate the database, then run the root user commands followed by the client commands in script order within each script file.**

Deliverables:
1. All of the .java files associated with your application.
2. All 14 screenshots from the execution of the commands specified in the **`project3rootuserscript.sql`** script.
3. All 8 screenshots from the execution of the commands specified in the **`project3clientuserscript.sql`** script.
4. A screenshot showing the final state of the **operationscount** table after executing the command `select * from operationscount;` once both the root user and client user command script files have been completely executed.

All should be uploaded to WebCourses no later than 11:59pm Sunday October 24, 2021. Be sure to clearly label each screen shot. Use the convention: RootCommand1, RootCommand2A, RootCommand2B, and so on. Similarly for ClientCommand1, ClientCommand2A, and so on.

Details:
Shown on the next page is a screen shot of the initial GUI. Notice that there are drop-down lists for selecting the JDBC driver and database URL that the client must select. The client must also specify a username and password (MySQL option) before connecting to the database.

You should provide buttons for the user to clear the command window as well as the result window. The status of the connection should be returned to the GUI and displayed in the connection area.

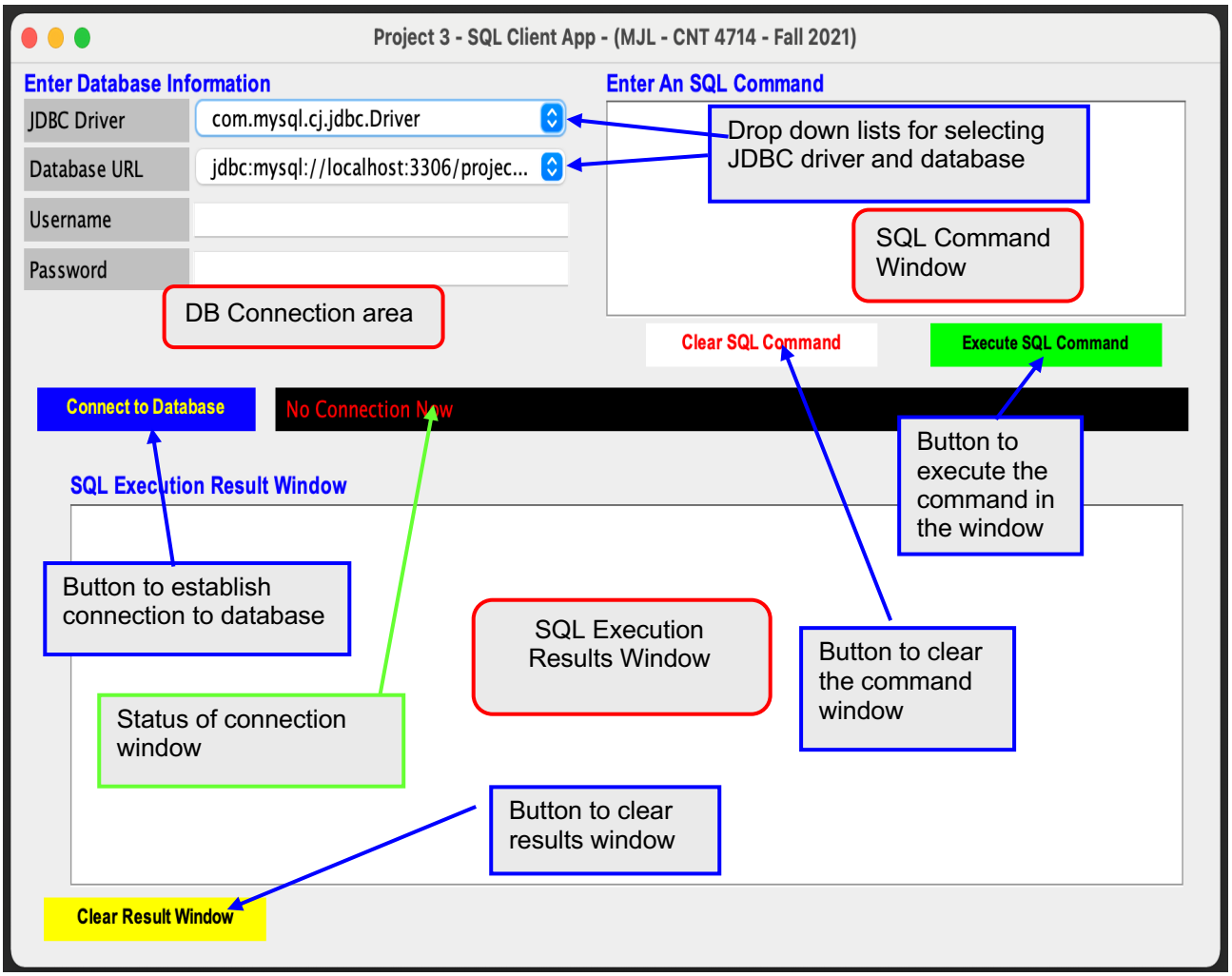
The output of all SQL commands should be returned to the SQL Execution Result window. Please note that only SQL commands can be executed via this application, we will not go to the effort of making the application display the results of MySQL-specific commands. (When a MySQL-specific command is executed, the SQL Execution Result window does not need to display any results, if you wanted to you could display the line "MySQL command executed" in the results window, but this is not required.)

As each user command is executed (only successful commands – some of the client command will not be successful) the **operationscount** table in the **operationslog** database must be updated by your application. Each query and each update will be logged (counted) separately. Your application must obtain a connection to the **operationslog** database and perform the update with root user credentials. Only successful operations will be logged – any transaction erroring will not increment any counter. These operations are invisible to the end user (regardless of who the user is, including root users). The application must connect to the **operationslog** database using a properties file which contains all necessary connection information.

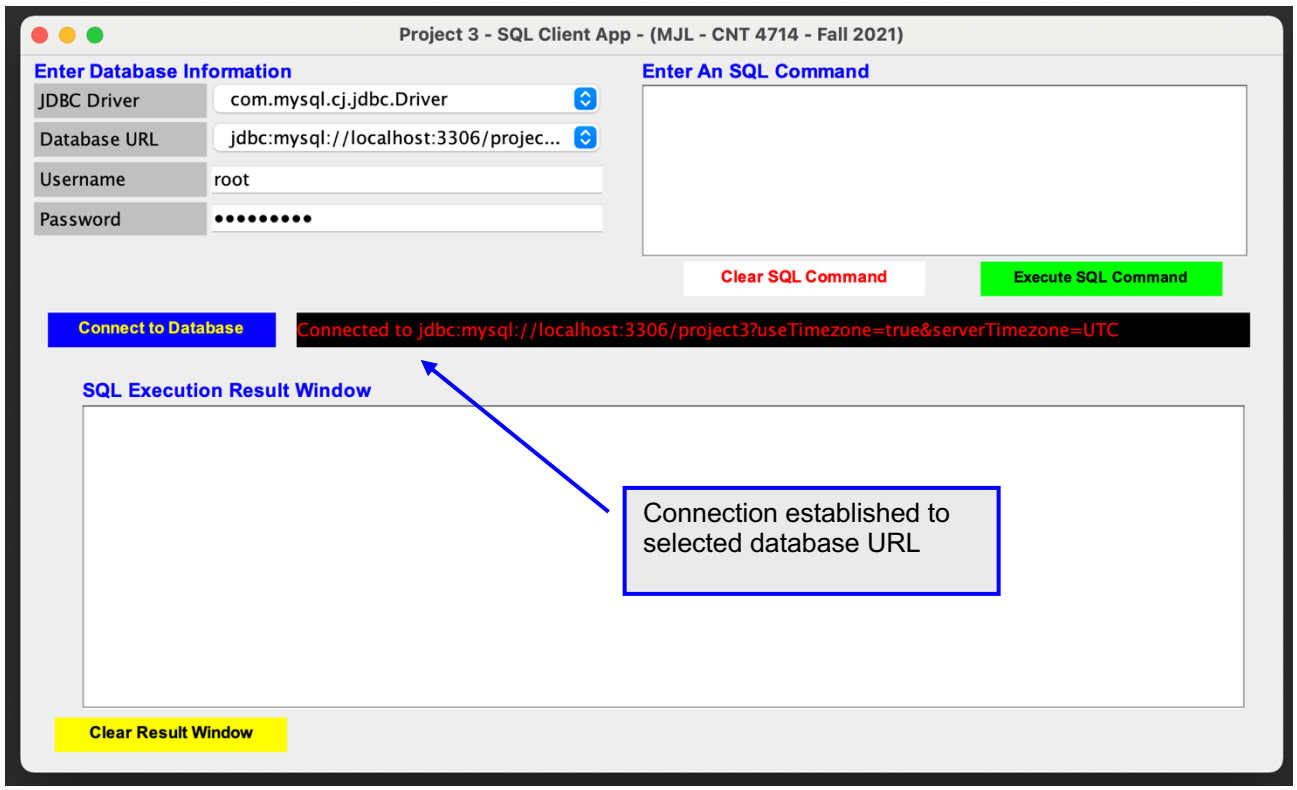
Note that for non-query DML and DDL commands, before and after screen shots must be taken to illustrate the basic effect of the command. See pages 8-9 for an illustration of this.

The remainder of the document illustrates the application at various phases during execution.

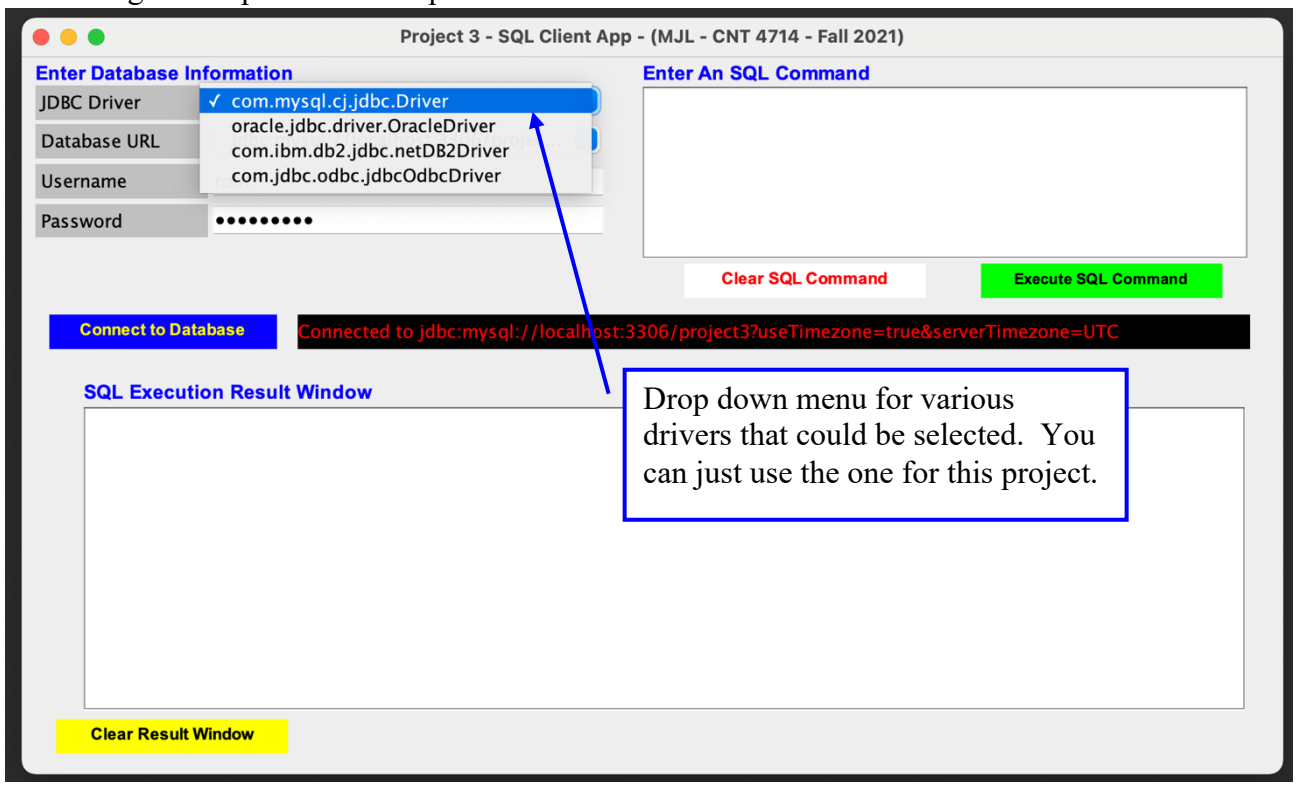
The GUI areas defined.



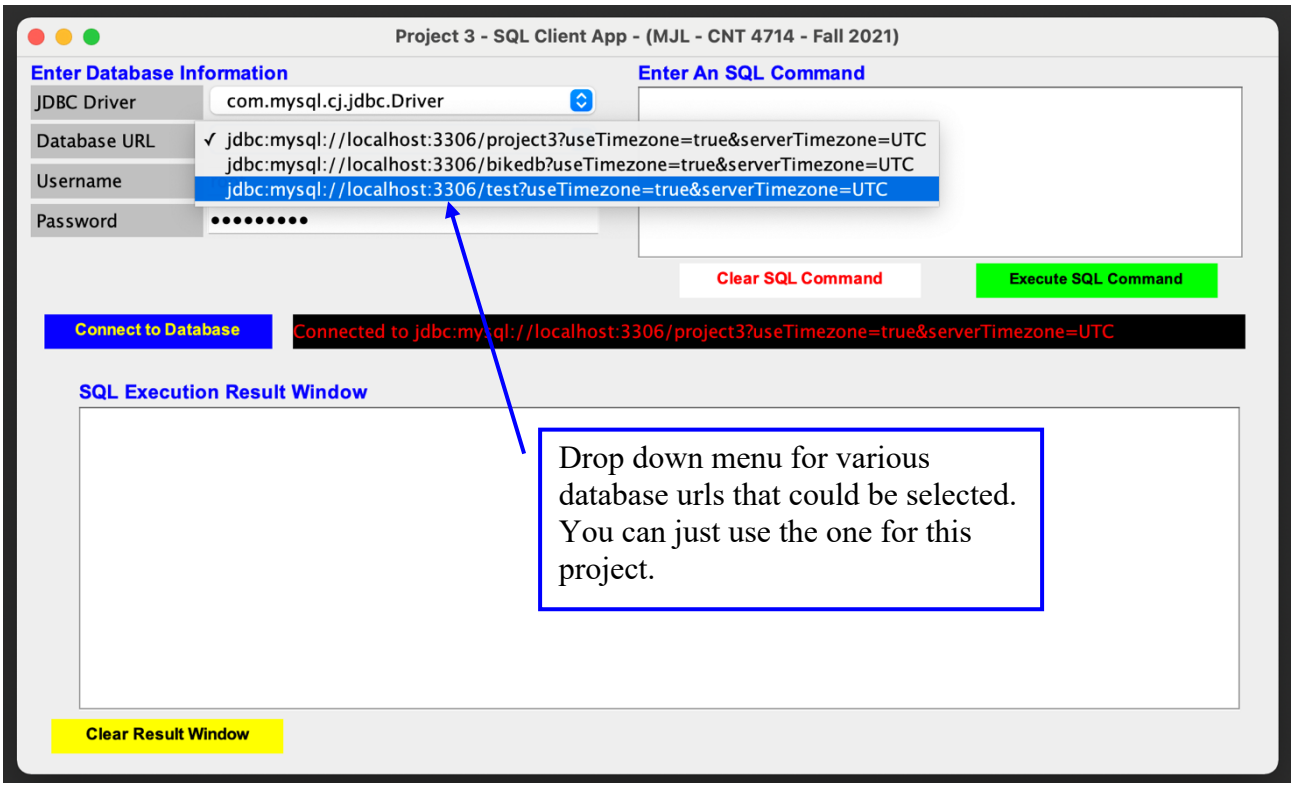
Screen shot illustrating an initial connection.



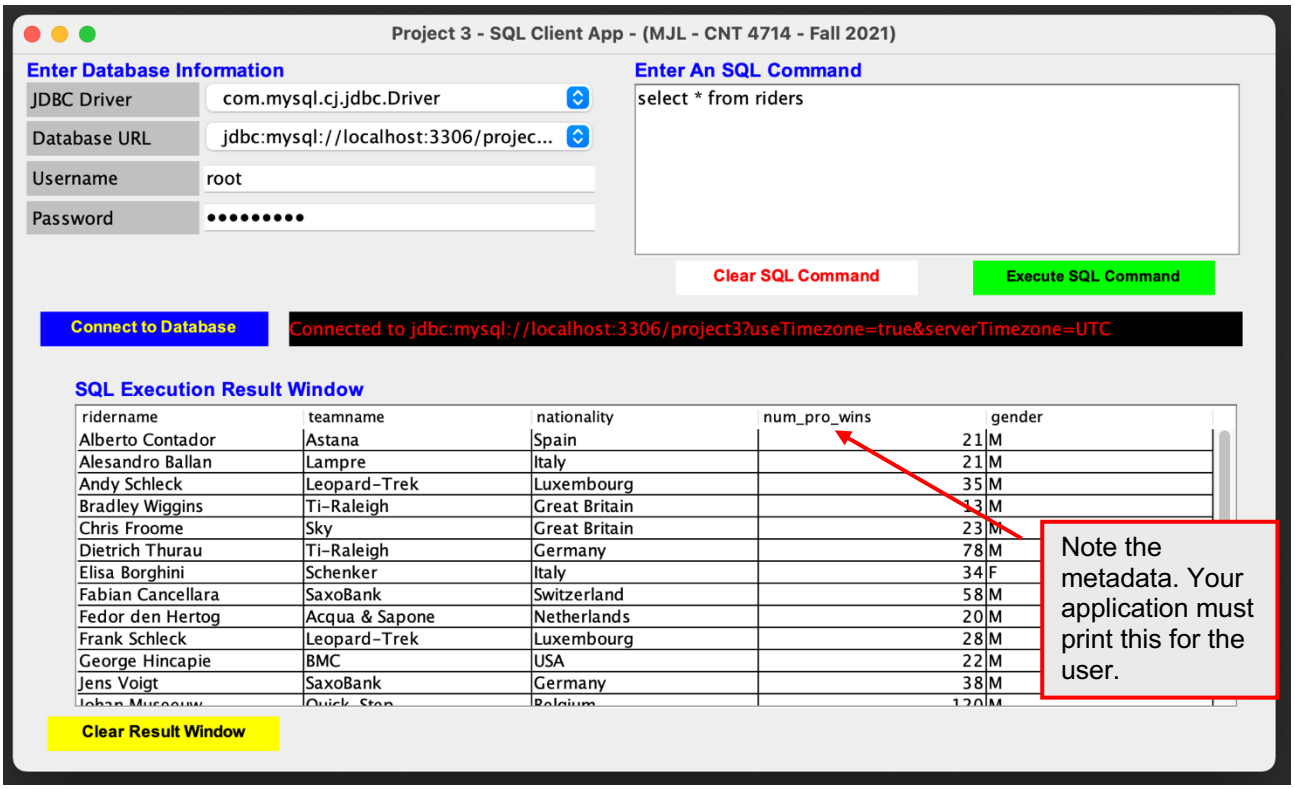
Illustrating the drop-down list of possible drivers that could be selected.



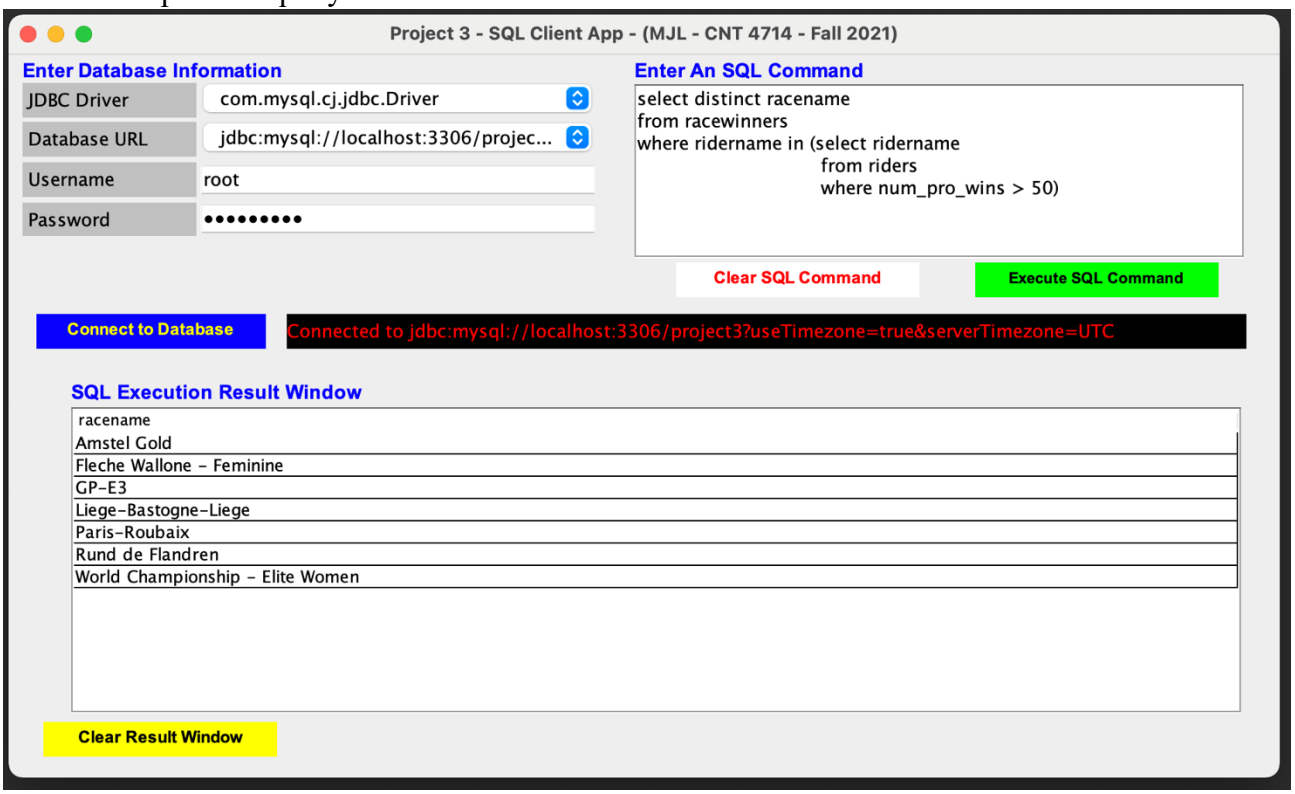
Illustrating the drop-down list of possible database URLs available.



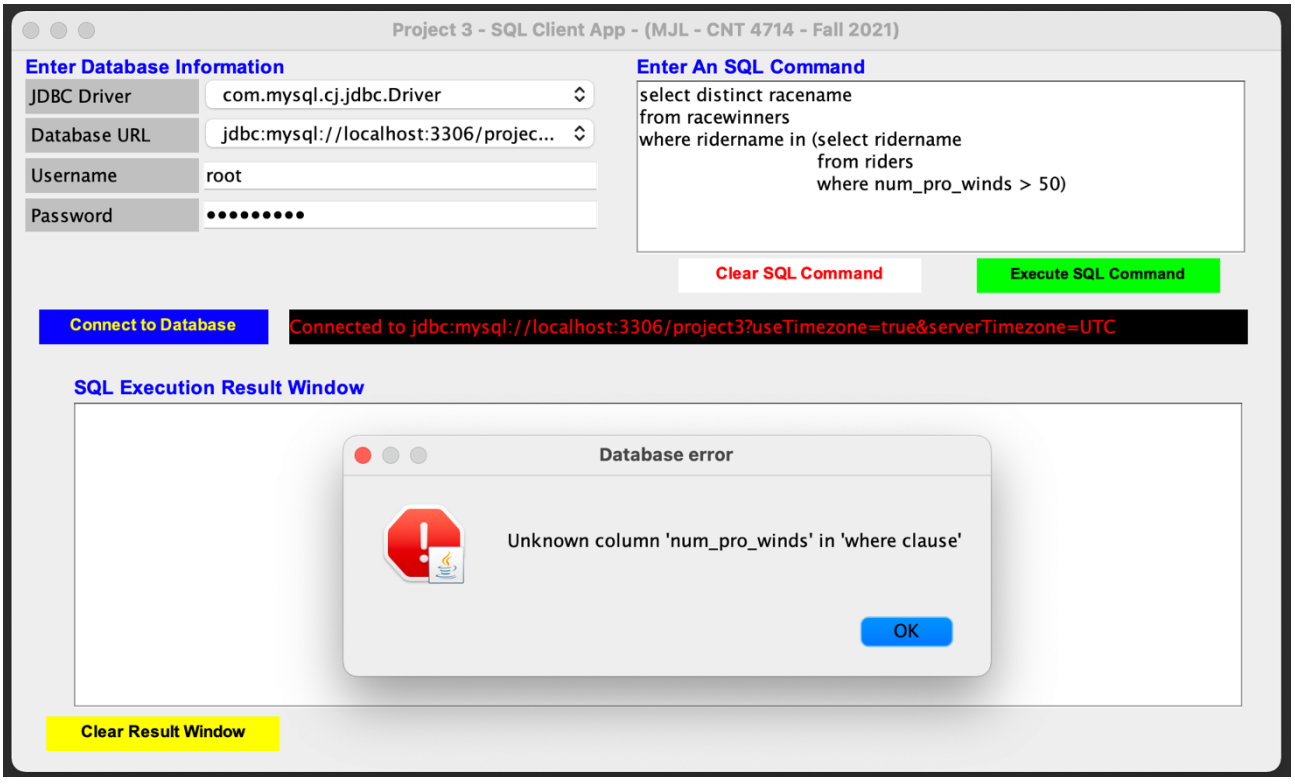
User has connected to a database and issued a select command. Results are displayed in the SQL Execution window.



A more complicated query:

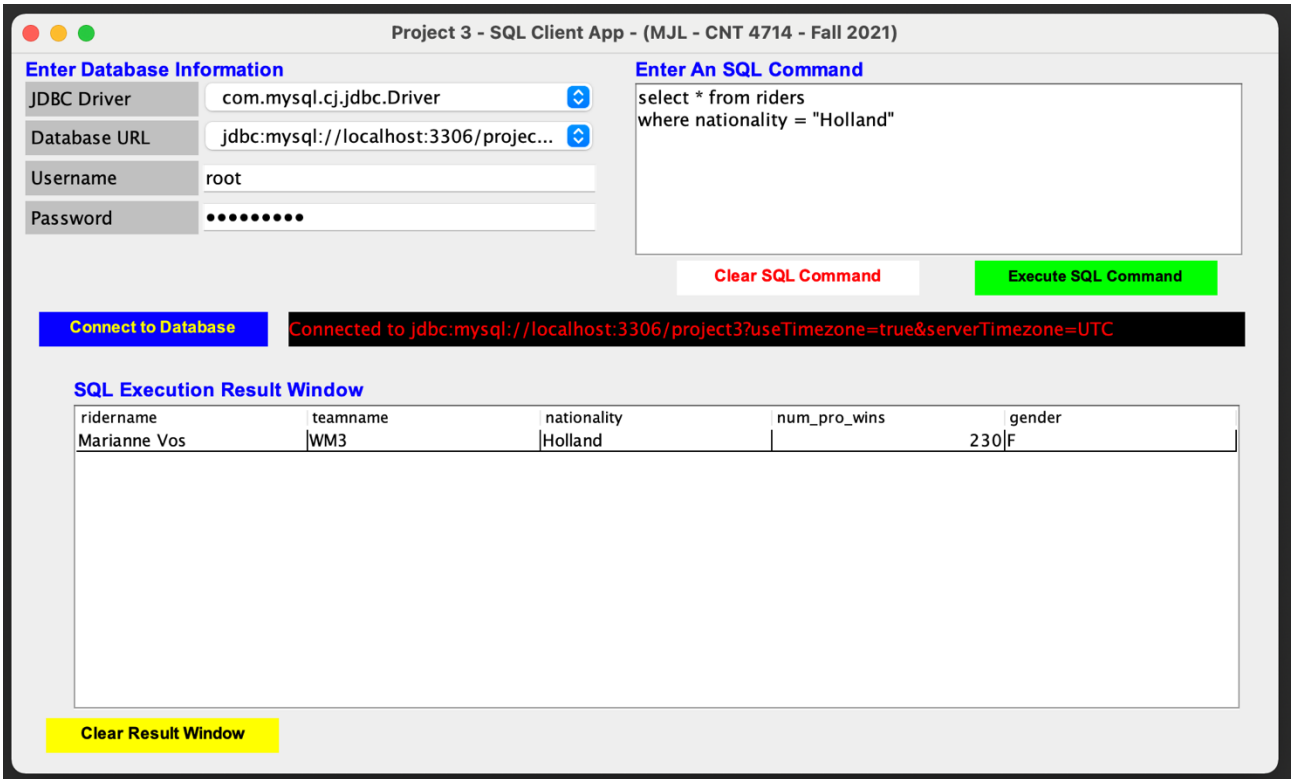


When the user makes a mistake entering a SQL command:

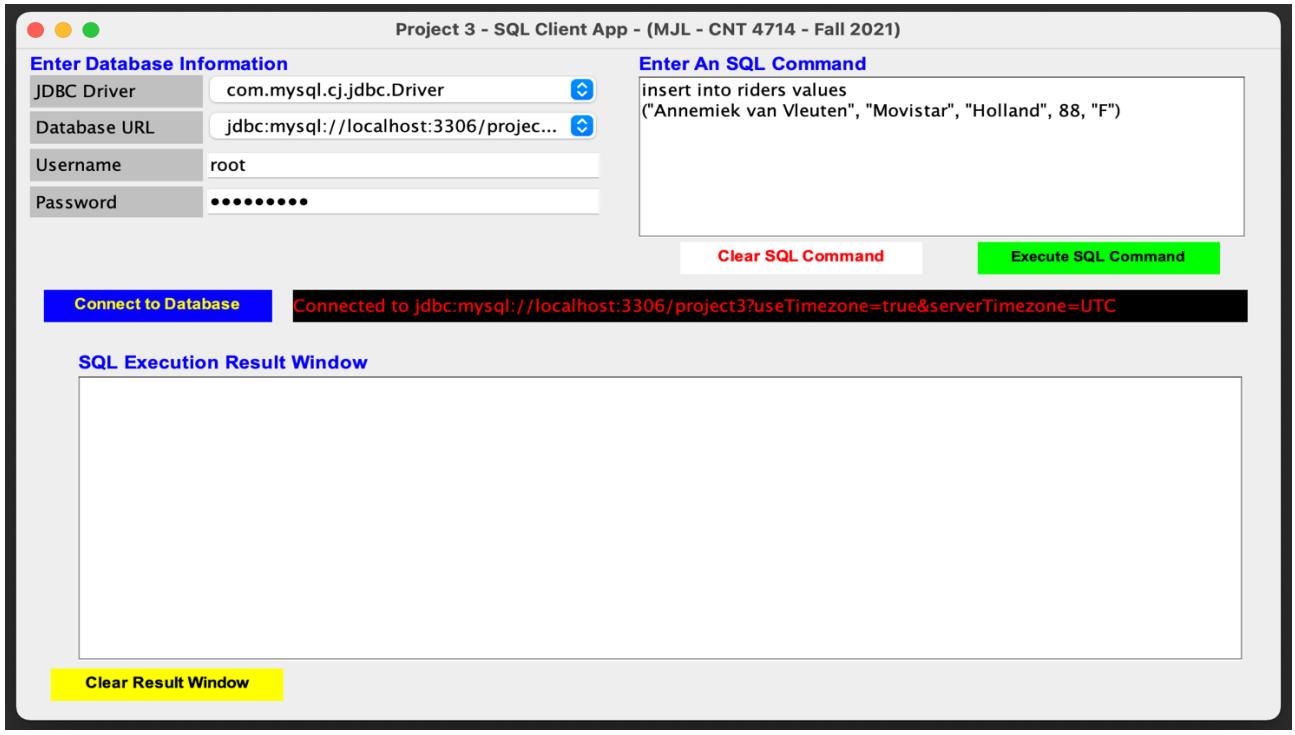


The following three screen shots illustrate that your application should be able to handle non-query commands from the users.

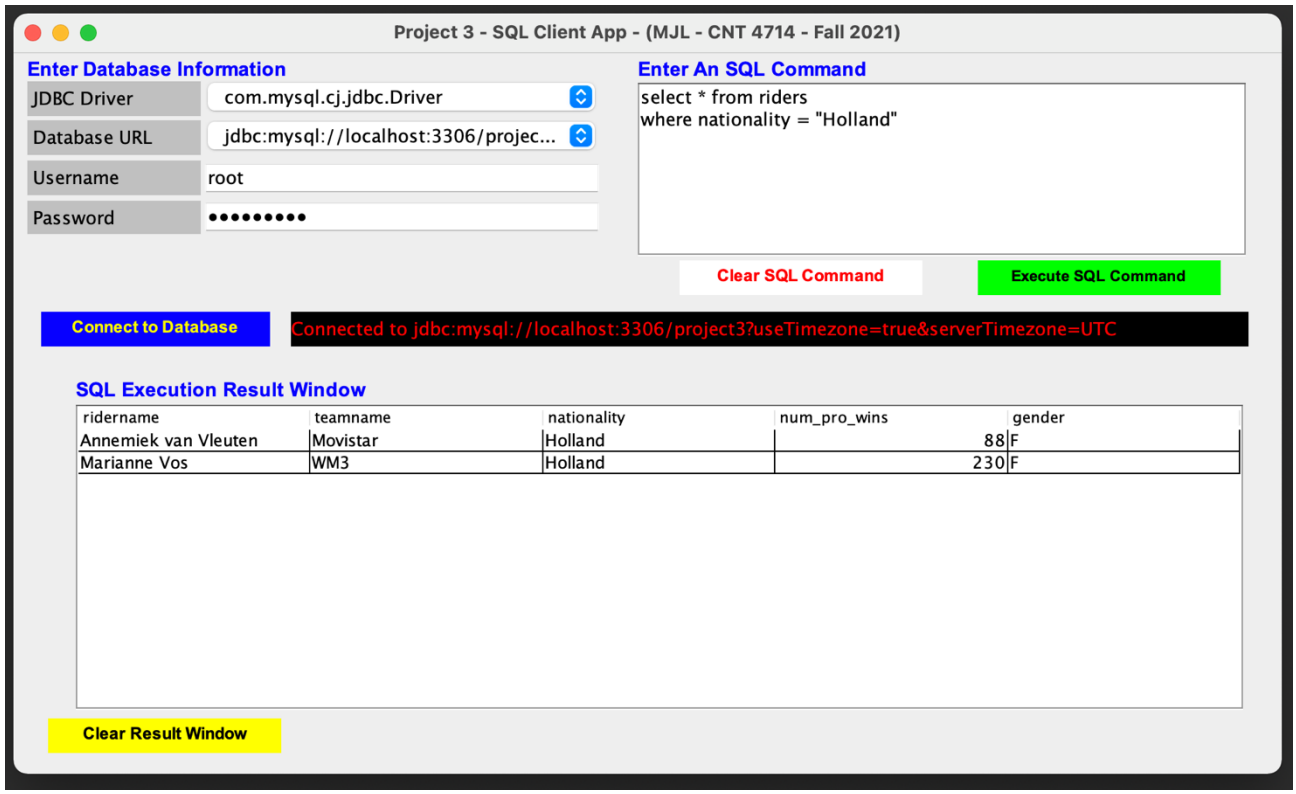
Before screen shot of a subset of the riders relation:



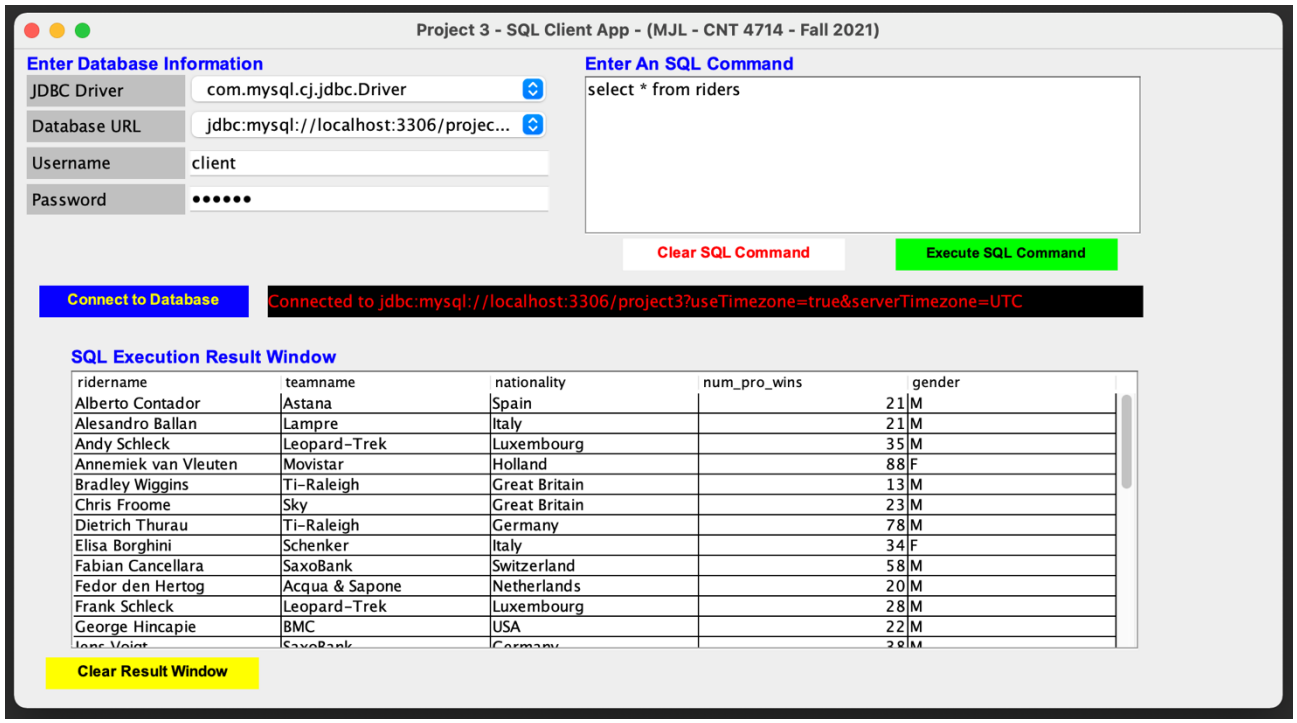
Insert command issued:



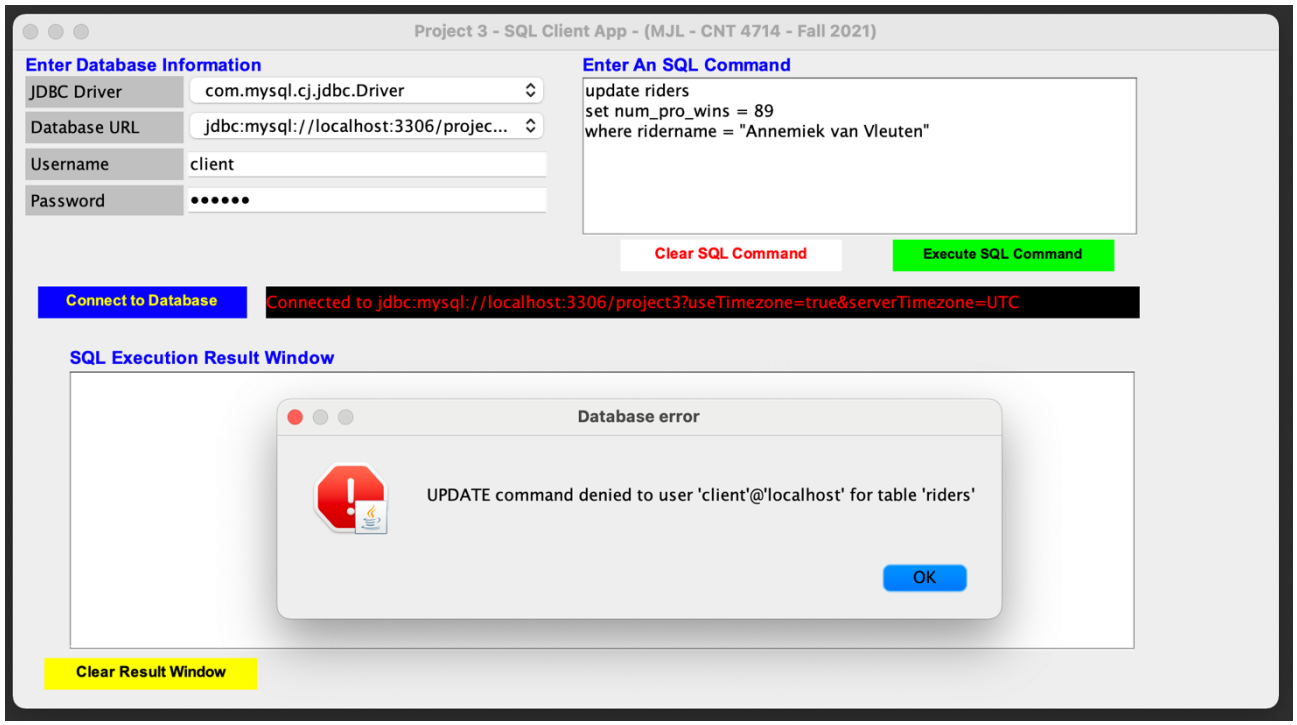
After screen shot of subset of riders relation after insert command was issued:



Screen shot illustrating the client user issuing a select command.



Screen shot illustrating the client user issuing a command for which they do not have permission:



The following screenshot illustrates the **operationscount** table values after various operations have been completed. This screenshot is taken from a root user account in the MySQL Workbench using the **operationslog** database. Note that the numbers shown in this screenshot are not the correct numbers that you will see after executing the root user command script followed by the client user command script. This is just an example.