**blackCSIS210 - Data Structures**

**Lab 1**

# Names: Michael Hamilton, Arizona Belden, Bailey Cross

# 

# General Lab Procedures

* You should create a directory (folder) in your home account called csis390. At the beginning of each lab, create a new sub-directory called labX, where X is the lab number.
* Files used in the lab can be found on the course canvas webpages.
* Turn in this lab sheet stapled to print outs of the code you produce as needed in each assigned section from the laboratory manual. These sheets should be in order. One lab submission is sufficient for each group.
* You can find documentation for the HTML 5 specification, the CSS, the Javascript, and php references respectively at

<https://html.spec.whatwg.org/>

<https://developer.mozilla.org/en-US/docs/Web/CSS/Reference>

<https://developer.mozilla.org/en-US/docs/Web/JavaScript>

<https://www.php.net/manual/en/langref.php>

**Lab Objectives**

* Practice using php
* Create and manage a table using SQL and MySQL

## Lab 9

#### Part 1 – Designing and making the users table

In this part of the lab, you will first design a table that will hold users information. Once accomplished, you will create a table using php code.

* Before creating a table with php, it is best to design the table appropriately. Just like designing a class in a high-level language, designing a table required appropriate fields with attribute datatypes.
* Let’s call our table UsersXXXXX, where XXXXX is the number you used in the previous lab for your Questions table.
* Which fields would you need for this table? Well, that depends on the needs of the application. For our trivia application, you would want to record information on a player’s username, a player’s password, the number of games the player has played, and the number of correct and incorrect answers (per game).
  + Which of these apply to a player’s table?

username, password, number of games

* + Which of these apply to a different table?

correct and incorrect

* List the field names for your table, along with their datatypes.

Players table: username(pk, varchar(12)), password(varchar(20)), number of games(integer(4))

Games table: game id(pk,autoincrement), correct(integer(2)), incorrect(integer(2)), username(fk)

* Should any of these have the ability to have a NULL value? If so, which ones and why? If not, why not?

Nothing should be null in this database, a value of zero would be acceptable if there are no games played or if the players had 0 correct or incorrect answers

* Which field or fields would be the primary key? Why?

username and game id because they will always be unique. No two users should have the same username and game id will be different since it is autoincrement

* Write your SQL statement to create a table called UsersXXXXX which will hold the necessary information and define a proper primary key.

CREATE TABLE Users51879 (

username varchar(12) PRIMARY KEY,

password varchar(20),

games\_played integer(4),

);

CREATE TABLE Games51879 (

game\_id INTEGER AUTO\_INCREMENT PRIMARY KEY,

correct integer(4),

incorrect integer(4),

user varchar(12),

FOREIGN KEY(user) REFERENCES Users51879(username)

);

* Create a php file called make\_users\_table.php which will create this table.
  + Make sure you use a URL key to protect this script so that others cannot execute it.
* Use WinSCP to connect to the server and upload your lab folder to the server
* Type the following URL into your browser:

http://<your location>/lab9/make\_users\_table.php?key=XXX

where XXX is the secret code you used to protect the script.

* This will create the table on the server. Verify that the table exists.
  + Write a show\_users.php script that shows the contents of the users table, using the techniques from previous labs.
  + Upload show\_users.php and run the script to verify the existence and the contents of the table.
* Notice how we use the URL parameters and the $\_GET variable to prevent others from running this code. You must know the key for the script to run.

#### Part 2 – Adding Users

In this part of the lab, you will add users to the table.

* For now, we are going to let anyone add users to the application. So, we will create a completely unprotected recursive script that generates a submission form and processes the submission form.
* In Notepad++, create a file called insert\_user.php.
* Follow the model from lab 8 to add the HTML template that includes links so you can use Bootstrap 4.
* In the body tag and above the script tags, add HTML code to create a form with input text fields to let some enter a username and a password.
* Be sure to include the form tag and set the following attributes:

method="get" and action="insert\_user.php"

* Be sure to give each form element a name, i.e., name="username" and name="password"
* The name of the submit button should be "action" and the value should be "Insert" i.e, name="action" and value="Insert"
* Add a block of PHP code to the top of the document to process the form.
* Use the $\_POST variable to save all the form values to PHP variables:

$username = $\_POST[username];

$password = $\_POST[‘password’];

* Using the insert user script as a model, write an if statement so that we will only perform the insert if the submit button was pressed and all the form elements are not blank, i.e. != ""
* Before you can Insert the user into the table, you must verify that the user is not already in the table.
  + You can use SELECT query to check whether the name exists in the table. If it does not, then the result would be empty.
* Once you have verified that the username is not taken, use PHP code to make and INSERT query into your UsersXXXXX table.
* Use require\_once to include your functions.php file
* Call your db\_connect function and execute the query as follows:

$mysqli = db\_connect();

$mysqli->query($sql);

$mysqli->close();

* Save insert\_user.php
* Use WinSCP to connect to the server and upload your file to the server.
* Type the following URL into your browser:

http://<yourlocation>/lab9/insert\_user.php

* Verify that all the fields are correctly inserted, using show\_users.php.
* Once completed, demonstrate your webpage for your instructor and have him initial here. If you do not finish during the lab period, then demonstrate your webpage at the beginning of the next lab period.

Instructor’s Initials \_\_\_\_\_\_\_\_

#### Part 3 – Script Integration

#### In this part of the lab, you will combine the previously written scripts into one code file.

* The two scripts you created can be unified into one combined script where after you insert a new question the table of questions is displayed so you can see that the insertion worked.
* Think of the insert script as a big if statement where you either (a) generate the form or (b) perform the insertion and then display all the user data. The key is that you also need to generate a hyperlink so you can go back to the form. When you click the hyperlink, the Insert button's value will be null and the form will be regenerated.
* Here you can simply cut and paste code from show\_users.php to insert\_users.php.
* Once completed, demonstrate your webpage for your instructor and have him initial here. If you do not finish during the lab period, then demonstrate your webpage at the beginning of the next lab period.

Instructor’s Initials \_\_\_\_\_\_\_\_

#### Part 4 – Processing an answer to a question

In this part of the lab, we will attempt to score a trivia question answer based upon a user’s response.

* Start with your get\_question.php script as a guide and save it as grade\_question.php.
* You will be using SESSION variables, so be sure that you start a session using session\_start();
* Store the question text in $\_SESSION[‘question’];
* Change the action attribute of the form to process\_question.php.
* Now, we will write process\_question.php.
  + Be sure to start the session in process\_question.php in order to gain access to the session variable.
* Use $\_POST[‘answer’] to obtain the choice sent from grade\_question.php.
* Write an SQL query to look up the answer value in the database for the question in the database table with the text stored in the session variable that we discussed earlier.
* If the answer is correct, echo a correct message to the screen.
* If the answer is incorrect, echo an error message to the screen, and state what the correct answer is.
* Once completed, demonstrate your webpage for your instructor and have him initial here. If you do not finish during the lab period, then demonstrate your webpage at the beginning of the next lab period.

Instructor’s Initials \_\_\_\_\_\_\_\_

#### Part 5 – Multiple questions on a page

In this part of the lab, we will create a trivia question page that allows up to X questions on a page.

* Using your code from the previous lab, create a new php script called get\_game.php, that has an entire game of questions on the screen at once.
  + The number of questions should be well ordered and formatted.
  + X should be at least five. Be sure that the number of questions in your question table is greater than five.
  + Use the rand() to display random questions from the table.
* **No submission needed. Grade will be based upon participation and the files uploaded to the servers.**