Introduction to Databases – Flipped Classroom FOUR

PHP, HTML and MySQL – NEED A LAPTOP FOR THIS!

General Instructions:

- You must have watched all the videos for Week 7 and Week 8, you must have completed the review exercises and completed Quiz 4 BEFORE doing the "flipped classroom" so that you are fully prepared to work on the exercises with your group.
- In your Owl group of students, TOGETHER as a group, discuss and complete the questions below on the paper.
- You will be given approximately 90 minutes to complete the questions (or as much time as it appears to be taking the majority of the groups to complete the questions).
- Once the 90 minutes are up, you will swap your paper with another group.
- We will then take up the questions together and you will write a score at the top of the paper and sign your group number as the markers and have one of your members sign the paper as the marker.
- Pass the marked sheet back to the original group
- Take your marked exercises and have one member in your group take a picture with his/her phone of the front page and upload at least ONE of your .php files. Make sure that for the first page, the group mark and marking groups signature is clear in the photo
- Have one group member upload and submit the images to owl for the flipped classroom ONE assignment AFTER IT WAS MARKED. This allows your group to use the flipped classrooms work for studying.
- The goal is for your group to learn from each other, so it is fine, actually encourage, to brainstorm and discuss and problem solve! Feel free to surf the internet or watch the course videos again to figure out your answers.

YOUR GROUP NUMBER: _____

Group Member Name (PRINT)	Present Today (Circle One)	
1	YES	NO
2	YES	NO
3	YES	NO

THIS AREA IS ONLY TO BE FILLED IN BY THE MARKING GROUP

Marking Group Number:	
Name of one of the marking group members (Printed)	
Signature of that group member	
SCORE OUT OF 1 (Total for this sheet is 40, so just divide by 40 and round to 1 decimal)	

Marking Instructions:

- The flipped classrooms are worth 2% each.
- Every group member who shows up to class and stays till the end will get 1% automatically.
- The other 1% is based on what you get on the questions below
- The 2 values are then added together to get your flipped classroom mark.
- Each question is worth 1 or 2 marks but has several parts, if the group got MOST of the question right, give them the full mark, if they got more than 50% wrong, give them half marks. Add the marks together at the end and give them a score out of 1 rounded to 1 digit. (e.g. 0.7 or 0.8 or 1.0)
- **DON'T BE TOO EVIL WHILE MARKING, the goal is to understand the concepts better while learning together, not get everything perfect (but if a group everything wrong, don't give them perfect-try to find a nice balance!)**
- Pass the marked sheet back to the group you were marking.

Objectives:

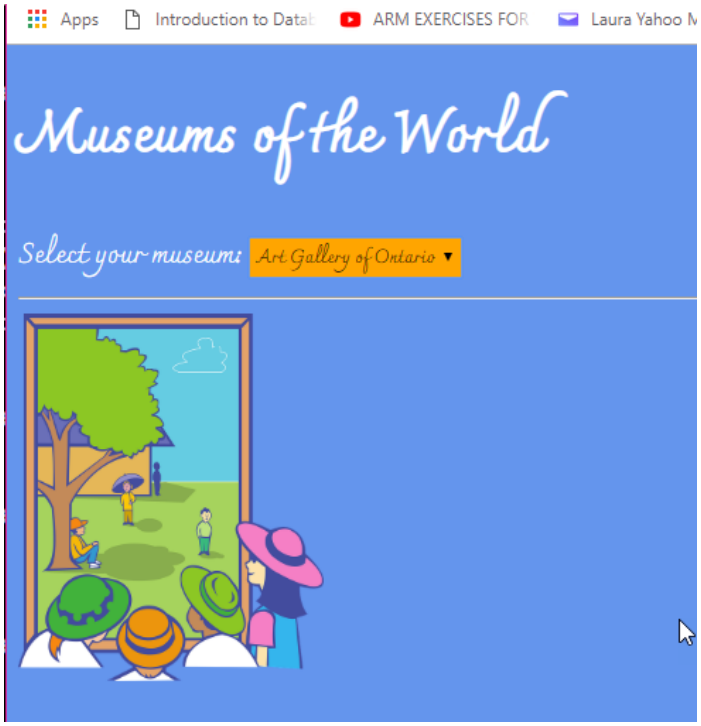
- To practice creating webpages that connect to a database
- To practice writing PHP code
- To learn collaboratively how to problem solve

QUESTION 1 – Creating the database and connecting to a table in the database (About 45 Minutes)

Part 1 (Creating the Database): You are going to create a museum database with 2 tables that have a one to many relationship (museum has many works of art). There will be a museum table and a works of art table. Museums contain works of art. Then you are going to create a webpage that connects to the database and displays the information and adds new works of art.

1. Have one member go into their virtual machine and then go into mysql as follows:
   `mysql -u root -p`
2. Go to: http://www.csd.uwo.ca/~lreid/blendedcs3319/flippedclassroom/four/dbsetup.txt and copy the SQL statements in this file and paste them into your virtual machine.
3. Do a SELECT on both the ***museum*** table and the ***workofart*** table to make sure you have data in your database.
4. Exit mysql
5. Move to the /var/www/html directory as follows:
   `cd /var/www/html`
6. Create a new directory called flipped4 as follows:
   `mkdir flipped4`
7. Move to that directory as follows:
   `cd flipped4`

8. Create a new file called ***museum.php*** in that directory. You can use nano or emacs or vi as your editor (depending on what you installed). If you want, you could create it on your machine in notepad or textedit and then copy and paste it, or you can work right in your virtual machine but keep in mind you won't have a backup of the file.
9. Put the html tags in museum.php to create a page that looks like this (see next steps for the code):



10. You can get the html code and css code that was used to create the page above OR if you want, you can write your own html and css to make something similar to the page above. If you want use our code to start you off, you can get the code here:
    a. HTML CODE: (do view source and copy the code to your *museum.php* file) http://www.csd.uwo.ca/~lreid/blendedcs3319/flippedclassroom/four/starter.html
    b. CSS CODE: (create a file in the *flipped4* directory called ***museum.css*** and put the following code in it): http://www.csd.uwo.ca/~lreid/blendedcs3319/flippedclassroom/four/museum.css
11. Go to the correct URL for the new page you just created and make sure that you can see it in a browser. It should be something like this (but change the virtual machine number): http://cs3319.gaul.csd.uwo.ca/**vm001**/flipped4/
12. Click on the *museum.php* file and make sure it works
13. Now we are going to write the code to connect to the database that you just created in step 2 above.

14. Create a new file called connecttodb.php in your virtual machine and put the following code in that file. Your group will have to figure out how to fill in the blanks below:

```
<?php
$dbhost = "localhost";
$dbuser= "root";
$dbpass = "__________";
$dbname = "___________";
$connection = mysqli_connect($dbhost, $dbuser,$dbpass,$dbname);
if (mysqli_connect_errno()) {
     die("Database connection failed :" .
     mysqli_connect_error() . " (" . mysqli_connect_errno() . ")"  );
    } //end of if statement
?>
```

Be careful here → if you cut and paste you MIGHT get the wrong type of quotes, so you might want to retype your quotes.

15. Go back and edit your ***museum.php*** and add the following code right after the <body> tag.

```
<?php
  include "con.php";
?>
```

16. I have purposely made a mistake on the middle line above because I want to make sure that your php errors are displaying. In the browser, reload the page again and make sure you get error messages. If you are NOT getting errors, go to Step 3 of http://www.csd.uwo.ca/~lreid/blendedcs3319/workshops/MySQLPHP.html to see if you can find a mistake (it might be on Step3 part 4, if you didn't modify your php.ini file) OR call your prof over to help you.
17. Change the middle line in step 15 to be:

    include "connecttodb.php";
18. Reload your page again and make sure you are not getting errors.

19. We have now established a connection to our database, now we want to retrieve some data. Create another new file called ***getmuseum.php***. Put the following code in that file but fill in the blank with the correct SQL statement to show all the data in the museum table;

```
<?php
  $query = "__________________";
  $result = mysqli_query($connection,$query);
  if (!$result) {
    die("databases query failed.");
  }
  while ($row = mysqli_fetch_assoc($result)) {
     var_dump($row);
     echo $row . "<br>";
   }
   mysqli_free_result($result);
?>
```

20. Go back to your ***museum.php*** file and after the </select> tag, add the following:

```
<?php
  include "getmuseum.php";
?>
```

21. Reload your browser. You should now see a dump of all the data from the museum table that looks pretty gross. We want to put the museum names in the dropdown box. So each time we get a museum row in our while loop, we want to surround it with <option> and </option>. Remove these lines:

```
var_dump($row);
echo $row . "<br>";
```

and replace it with a new echo line (or lines) that will display the tag ***<option>,*** then the contents of the *musname* column (you would do $row["musname"] to get that value), then a ***</option>*** tag.

22. Save your ***getmuseum.php*** file and reload your page and make sure that you can now see the museum names. The probably are NOT in the dropdown box because in step 20 we put the 3 lines of code into ***museum.php*** AFTER the end of the dropdown box. They should be before the </select>. Try moving those 3 lines above the </select>, save your file and reload your browser. Make sure the museums are now in the dropdown box.
23. Notice that we still have the *Art Gallery of Ontario* even though it is not in our database. Go the museum.php file and change the *Art Gallery of Ontario* to just say *"Select Here".*

24. Reload your file in the web browser and view the source (the html). NOTICE THAT YOU CANNOT SEE THE PHP CODE, only the html tags that you echo commands displayed.
25. Modify your echo statement so that it not only displays the museum name but it also puts the museum's id into the value statement for the <option> boxes because you will need that for the next step. For example, one of your option html tags should NOT be this anymore:
    `<option>National Museum of China</option>`
    but instead should be this:
    `<option value='4'>National Museum of China</option>`
    NOTE: the mysql column names are case sensitive so use musID NOT musid
26. Save your ***getmuseum.php*** file and view the source to make sure that it is putting the value on the <option>
27. Now we want to make our code display all the works of art depending on the item that the user selects from the dropdown menu.

QUESTION 2 – Creating a Join (About 1 hour)

1. The tricky part is that AFTER we load our page, we have already gone to the web server (your virtual machine), got the data we needed and now we are back at our page on the client. So now the user will pick one of the museums and we need to go back to the database to get all the art work for the museum the user picked. The only way to get art work data is load up another .php page or reload our page, both of which force us to go to the web server again to get more data. In the workshop you did last week, you called another php page, in this example we will try reloading our page. It is a bit tricky where you put the code.
2. First let's create a javascript file that listens for the user clicking on the dropdown box (ie. Selecting a museum from the dropdown box). On your Virtual Machine, in the same directory, create a new file called ***museum.js***.

3. In the ***museum.js*** file add the following code:

```
window.onload = function() {
   prepareListener();
}

function prepareListener() {
    var droppy;
    droppy = document.getElementById("pickamuseum");
    droppy.addEventListener("change",getArt);
}

function getArt() {
  alert ("Hello world");
}
```

   This code is adding an action listener to the dropdown box that listens for the dropdown box to be changed. When it is changed, it calls the ***getArt()*** function.

4. Try reloading your code and seeing if it works (does it say Hello World?). It probably won't because of 2 missing things in your ***museum.php*** file:
   a. You need to put a <script> tag in ***museum.php*** so that it knows about this new javascript file we have created. Figure out how to properly insert the correct script tag just after the <body> tag to point to your new museum.js file
   b. You need to put an ***id*** on the <select> tag in ***museum.php*** so that the addEventListener knows which part of your page it should be listening for. Figure out what name to give the id attribute (HINT: look at your .js file) and put that id attribute on the <select> tag in ***museum.php***
5. Save your ***museum.php*** and reload it and make sure that when you change something in the dropdown, that "Hello World" pops up.
6. Now we are going to create a form AROUND our <select> tags in ***museum.php*** so that we can send the value for the museum that the user selects (from the <option> tags) to a form (i.e. BACK up to the webserver). Edit museum.php and add this line right before the opening <select> tag:

   <form action="" method="post">

   and add this line right after the closing select tag → </select> tag>

   </form>

   Notice for the <form> tag that we put ***action=***""If you put "" with NO .php file between the double quotes, then it reloads the current page. So ***action=***"" will reload this page. It is actually easier to just load a new page. For your assignment 3, you will likely find it easier to put a new webpage between the quotes, e.g. action="loadTheInfo.php" . I had to screw around a fair bit to do this next part and make it go up to the server AND stay in the same page ☹!

7. Even though we put an id on the <select> tag, forms don't use ids, they use the ***name*** attribute, so change your <select> tag so that is looks like this:
   <select name="pickamuseum" id="pickamuseum">
   The form will send the value to the server for anything in between <form> and </form> that has a name, so that is why we added the name to the <select>. The form is going to return the value for the selected option, i.e. the primary key for the museum that the user selected.
8. Now that we are sending the primary key for the selected museum back up to the server, we need to write the SQL to join with the works of art table and display all the works of art at the selected museum. For example, something like this:
   SELECT * FROM workofart WHERE whichmus = 3;
   The 3 comes from the value attribute on the <option>. It gets set when the user picks one of the options in the dropdown box. We will add some php to do take the value and create the appropriate SQL command. Create a new file called ***getartwork.php*** on your virtual machine.
9. First of all, we need to get the value for the selected museum. In this new file, put the following 2 lines and for the third line, fill in the blank for the query to get a query that will return all the works of art for the selected museum similar to the query in the previous step:

```
<?php
    $whichMus = $_POST["pickamuseum"]; //get selected museum value from the form
    $query = _____________________________________;  //fill in with correct query
```

10. Now add the following lines below the 3 lines above:

```
   $result = mysqli_query($connection, $query);
   if (!$result) {
     die("databases query on art pieces failed. ");
    }
    echo "<ul>";   //put the artwork in an unordered bulleted list



    echo "</ul>";   //end the bulleted list
    mysqli_free_result($result);
?>
```

11. Save your file but stay in that file (don't close it, keep editing it).

12. Next, you need a while loop to walk through each of the rows returned by the $query. The while loop should go between the <ul> tag and the tag (these tags create an unordered or bulleted list). Look at step 19 in Question 1 above to see how to get at each row in a while loop and put the result for each row in the variable $row. Within the loop, you need to put the html list tags <li> and tags around the name of each of the works of art (column name: artname) and the artist name (column name: artist) and put the word BY in between the art and the artist. You want the display to look something like this→
Add the code to do that, save your file.

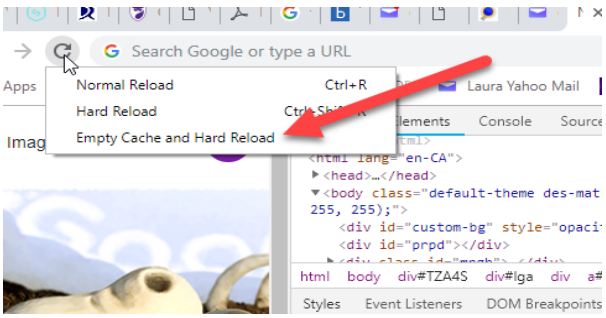
13. Now we need to call this code from our ***museum.php*** file. Edit that file and add code below where you added the </form> tag and right after the first <hr>. Since you are going back up to the server again, you will need to connect again AND you will need to call ***getartwork.php***, so you need to add the lines in red:

```
</form>
<hr>
<?php
    include "connecttodb.php";
    include "getartwork.php";
?>
```

14. Save your file and reload it. When you select a museum it is probably still printing "Hello". You need to fix your javascript files (***museum.js***)for this. Go into it and removed the alert ("Hello World") line and add a line that resubmits this page back up to the webserver as follows:

```
function getArt() {
    this.form.submit();
}
```

NOTE: when you reload here, your browser might NOT clear the cache so you will be using the old 'Hello World' version of the javascript. To clear the cache in Chrome, you need to right click and open the ***Inspect*** window, then hold down the reload button and you should see this (look at red arrow in image), make sure you pick:
Empty Cache and Hard Reload:



15. Now reload again, still one problem. The very first time we load the page, the user won't have had a chance to pick a museum yet. We can check if this is the first time the page has been submitted by modifying the ***museum.php*** file as follows (only display the works of art if they have picked a museum) by wrapping the following *if* statement around it.

```
</form>
<hr>
<?php
    if (isset($_POST['pickamuseum'])) {
      include "connecttodb.php";
      include "getartwork.php";
    } //end of if
?>
```

16. Try reloading again, now everything should work!

QUESTION 3 – Creating Users (About 15 Minutes).

Now you are going to write a page to check the password of a user before they can look at your database.

1. View the source for this webpage: http://www.csd.uwo.ca/~lreid/blendedcs3319/flippedclassroom/four/checkuser.html and copy the source into a new file on your virtual machine called ***checkuser.php***.
2. Look at the html tags for the above page you just created and figure out what name to give the php webpage that will check the user's id and password (HINT: look for the form tag)
3. Create a new PHP page and give it the name you figured out in step 1 above.
4. In this new file add the following lines:

```
<?php
   $dbhost = "localhost";
   $dbuser=  _____________________;
   $dbpass = $_POST['pwd'];
   $dbname = "flipped4db";
   $connection = mysqli_connect($dbhost, $dbuser,$dbpass,$dbname);
   if (mysqli_connect_errno()) {
        die("Database connection failed checking user");
   }
?>
```

5. Figure out what should go in the blank line above. HINT: I purposely made a mistake (left something out) in the file in step 1. You need to fix that problem in ***checkuser.php*** and then fill in the blank above to make this work.
6. Save both your files and reload the checkuser.php file and make sure that it fails if an incorrect password is added.

7. Notice that if you supply user *root* and the correct password, it does nothing. You need to tell it which page to go to if the password works. To do that, open your password checking file and add the red lines below.

```
    if (mysqli_connect_errno()) {
        die("Database connection failed checking user");
    } else {
        header('Location: museum.php');
    }
    exit;
?>
```

8. These lines say: go to the location of the file called ***museum.php*** and then exit from this page. Save your file and retry to make sure it goes to the museum page if the password was correct.
9. Now let's try adding a new user and see if we can make it work for someone other than the root user. Get into mysql
10. Create a new user called ***homer*** and give ***homer*** the password ***bart*** as follows:

```
CREATE USER 'homer'@'localhost' IDENTIFIED BY 'bart';
```

11. Reload your ***checkuser.php*** webpage again and see if it works when you type in ***homer*** with a password of ***bart***. It probably fails because even though you have created a new user, you haven't given your new user any privileges.
12. Go back into mysql and figure out how to allow homer to view all the data in the flipped4db database. HINT: here is the start of the command you will need:

```
GRANT ____ ON ____ TO '___'@'localhost';
```

13. Reload your ***checkuser.php*** webpage and make sure that homer can see all the museums and the works of art for each museum.
14. RECAP: If you want to validate a user you need to:
    a. Create a password webpage
    b. Create a user in mysql with a password
    c. Grant the privileges you want the user to have in mysql

QUESTION 4 – Inserting data into tables and debugging (About 20 Minutes).

In this next section, you are going to write code that adds a new work of art to a museum. I will give you ALMOST WORKING code with a few errors; you need to find the errors.

1. Go to this page: http://www.csd.uwo.ca/~lreid/blendedcs3319/flippedclassroom/four/addnewartwork.html view the source, copy the source, and paste it into a new file called ***addnewartwork.php*** on your virtual machine. Save this file and exit from it.
2. When you submit the new page you just created, it will call a page named ***addthework.php.*** Create a new file called ***addthework.php*** and put the following code into it:

```
<?php
   include 'connecttodb.php';
   $art = $_POST["work"];
   $artist = $_POST["artist"];
   $theyear = $_POST["theyear"];
   $whichmus = $_POST["whichmus"];

   $query= 'INSERT INTO artwork (artname, artist, year, whichmus)  VALUES ("' .
$art .'","' . $artist . '","' . $theyear . ',' . $whichmus . ');';

   if (!mysqli_query($connection,$query)) {
      die ("Error while trying to add new art". mysqli_error($connection));
   } else {
     header('Location: museum.php');  //send back to museum page once it is done
     exit;
  }
?>
```

3. Now, in your browser, go to the directory we are putting all the files in on your virtual machine, it will be something like: ***http://cs3319.gaul.csd.uwo.ca/vm???/flipped4*** and click on the page called ***addnewartwork.php***. Enter some data and see if it works (it should NOT work). Use echo commands to print out what is in the variables and to double check your $query and see if you can find the mistakes. There are 3 mistakes you need to fix.

QUESTION 5 – Triggers (About 15 Minutes).

If you have time, create a trigger on the ***workofart*** table inside of mysql that is triggered whenever a new work of art is added. If the new row to be inserted has a null value for the artist or has a value of "" then set the artist to be "Unknown". To refresh your memory, here is an example of a trigger:

--

```
mysql> delimiter //
mysql> CREATE TRIGGER upd_check BEFORE UPDATE ON account
    -> FOR EACH ROW
    -> BEGIN
    ->     IF NEW.amount < 0 THEN
    ->         SET NEW.amount = 0;
    ->     ELSEIF NEW.amount > 100 THEN
    ->         SET NEW.amount = 100;
    ->     END IF;
    -> END;//
mysql> delimiter ;
```

Use Question 5 above to test your trigger by adding a new piece of art where the artist is empty. You should NOT have to change or touch the php code if you did your trigger correctly. Then go to ***museum.php*** to prove that it put ***Unknown*** for the artist's name.

Show your website with the insert part working to the group next to you and get them to sign your sheet. Then upload at least some of the code (one of your .php files) into owl AND WRITE DOWN THE NAMES OF THE GROUP MEMBERS WHO SHOWED UP TODAY IN THE OWL TEXT BOX AND HIT SUBMIT