

COMP2245 - Lab 5 (20 marks)

PHP and SQL

Due Date: Thursday, November 17, 2022

For this lab we will develop a simple Web service in PHP that we can query and retrieve results using SQL. You will also need to install XAMPP or MAMP which will come with Apache, MySQL and PHP installed together. Links to download either can be found below:

XAMPP: <https://www.apachefriends.org/index.html>

MAMP: <https://www.mamp.info/>

Here is a short video on how to install XAMPP on Windows: <https://youtu.be/FiFJ1jTfITc>

You will be required to submit the URL of your Github repository by the deadline specified above.

At the end of this lab you will have done the following:

1. Setup Databases
2. Try some SQL queries
3. Create a basic PHP script that uses SQL with request variables
4. Integrate that PHP service into an AJAX system (similar to the exercise from lab 4)

Overview

We will be creating a service called **world.php** that serves information about countries of the world. Pass it a country name as a parameter in the *Query String* of the URL, and it outputs some basic statistics about that country. For example, for the term "Afghanistan":

```
world.php?country=Afghanistan
```

Create your Repository

You are also required to create a Git repository on Github called **comp2245-assignment5** and after each exercise commit your code and push to Github. So let us start by creating that repository.

1. You can create a new Github repository by going to <https://github.com/new> or by clicking the "+" icon in the top right once you are logged into the website and select "New Repository".
2. For this lab, use the repository name as **comp2245-assignment5**.
3. Ensure that your repository is *Public* and that you select the option to initialize the project with a README file.
4. The remaining options can be left at their defaults. Then click on **Create Repository**.

Search or jump to...7


Pull requestsIssuesMarketplaceExplore

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner ^{*}

Repository name ^{*}

 uwi-info2180

 /

info2180-lab1

✓

Great repository names are short and memorable. Need inspiration? How about [fuzzy-octo-palm-tree?](#)

Description (optional)

☒ Public

Anyone on the internet can see this repository. You choose who can commit.

☐ Private

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☒ Add a README file

This is where you can write a long description for your project. [Learn more.](#)

☐ Add .gitignore

Choose which files not to track from a list of templates. [Learn more.](#)

☐ Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)


This will set [master](#) as the default branch. Change the default name in [uwi-info2180's settings.](#)

Create repository

5. On your newly created repository, click on the "Code" button and copy the URL in the box that appears.

<> Code🕒 Issues🔗 Pull requests🔄 Actions📁 Projects📖 Wiki🛡 Security📊 Insights⚙ Settings

🔗 master1 branch0 tags

 ylynfatt Initial commit

📄 README.md

Initial commit

README.md

info2180-lab1

Go to fileAdd file↓Code↓

📄 Clone

HTTPS SSH GitHub CLI

https://github.com/uwi-info2180/inf

📄

Use Git or checkout with SVN using the web URL.

🖱 Open with GitHub Desktop

📄 Download ZIP

About⚙

No description, website, or topics provided.

📖 Readme

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

© 2020 GitHub, Inc.

TermsPrivacySecurityStatusHelp

Contact GitHubPricingAPITrainingBlogAbout

6. Open your command prompt, navigate to the **htdocs** or **www** folder for your installation of XAMPP, MAMP or WAMP (e.g. using the `cd` command) where you would like to clone your repository and using the URL you copied in step 5, type the following:

```
$ git clone https://github.com/<your-username>/comp2245-assignment5.git
```

NOTE: Ensure you change the URL and use the one that you copied in Step 5 with your username.

7. Then navigate to your newly cloned repository by typing:

```
$ cd comp2245-assignment5
```

8. You can then type the command `ls` and you should see a `README.md` file.
9. In the file add:

```
# COMP2245 Assignment 5
```

```
This is Assignment 5 for <Your Name> on PHP and MySQL
```

Of course, change `<Your Name>` to your actual name.

10. Save the file and then at the command line (or Windows Powershell), type:

```
$ git add README.md  
$ git status
```

You should then see that the file has been staged.

11. Commit these changes to your local Git repository:

```
$ git commit -m "Edited README.md"
```

12. Check the status of your Git repository:

```
$ git status
```

It should now mention that there is nothing new to commit and that the working tree is clean.

13. Great! Now we should push this information to GitHub.

```
$ git push
```

It may ask you to enter your Github username and password. Enter it and then press enter. If all goes well you can then view your Github repository on the Github website and you should see that the file exists on Github with the changes you made to the file.

14. To see the history of the commits you can run the following command:

```
$ git log
```

15. Next, Create a new branch called **php-mysql-implementation**.

```
$ git checkout -b php-mysql-implementation
```

16. Now download the starter files (if you have not already done so) for the lab at the following URL:

<https://github.com/uwi-comp2245/comp2245-assignment5/archive/master.zip>

17. Unzip and copy the starter files (world.sql, world.php, index.html, world.css) from the link above into your **comp2245-assignment5** folder that you cloned in Step 6 and do an initial commit.

```
$ git add .
```

```
$ git commit -m "Added initial starter files"
```

18. Now begin the next set of exercises.

Exercise 1 - Setting up the "world" database

MAMP, XAMPP and WAMP all come with a database tool called PHPMyAdmin which allows you to connect to, import a database (and its respective tables) and also run queries on the database. In your PHPMyAdmin you will need to upload the file `world.sql`, from the starter code. Ensure you look through this file to see some of the SQL statements that will be run to create the database.

With your MAMP or XAMPP Server already installed, ensure that the Apache and MySQL Service is started. Then to load the MySQL database with the **world** database bring up PHPMyAdmin by browsing to:

<http://localhost/phpmyadmin> (XAMPP and MAMP on Windows)

or

<http://localhost:8888/phpmyadmin> (MAMP on MacOS)

Note: depending on how your installation of XAMPP, WAMP or MAMP was setup you may need to change the port on the end.

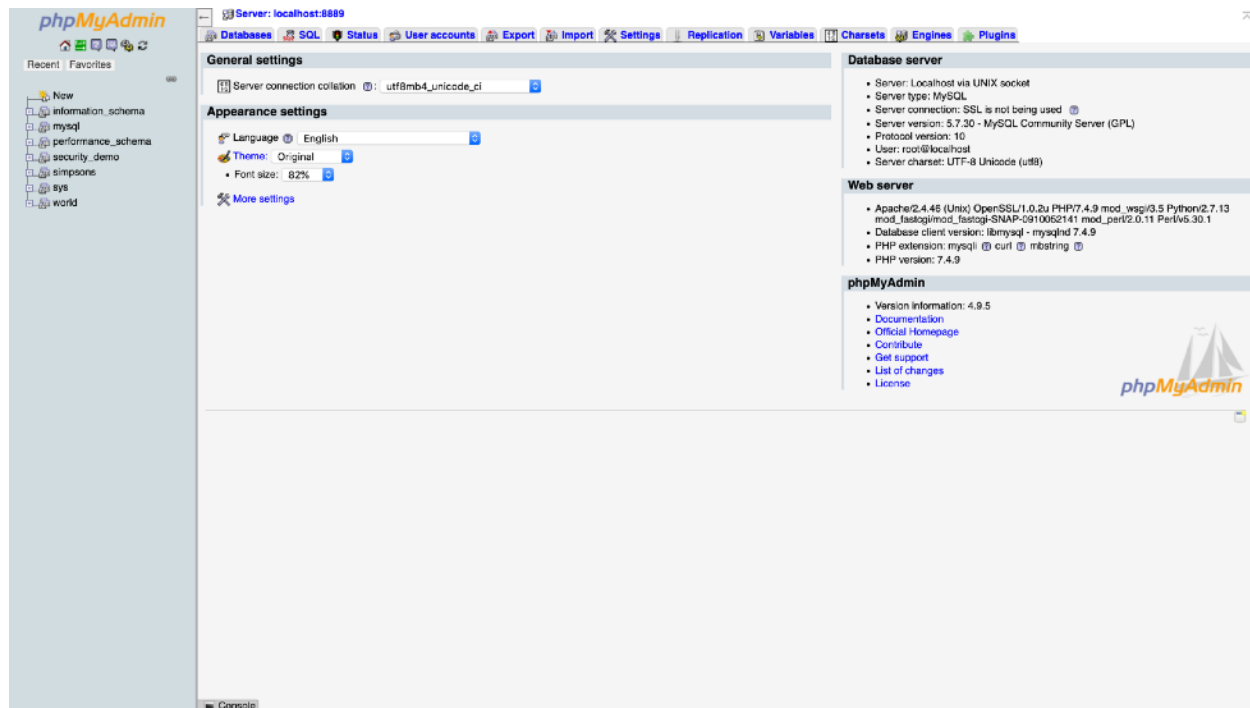


FIGURE 1: PHPMYADMIN SITE

Then click on "Import" and then under the "File to Import" section, click "Browse" and select the world.sql file from your comp2245-assignment5 folder. Next click on "Go"

at the bottom of the page.

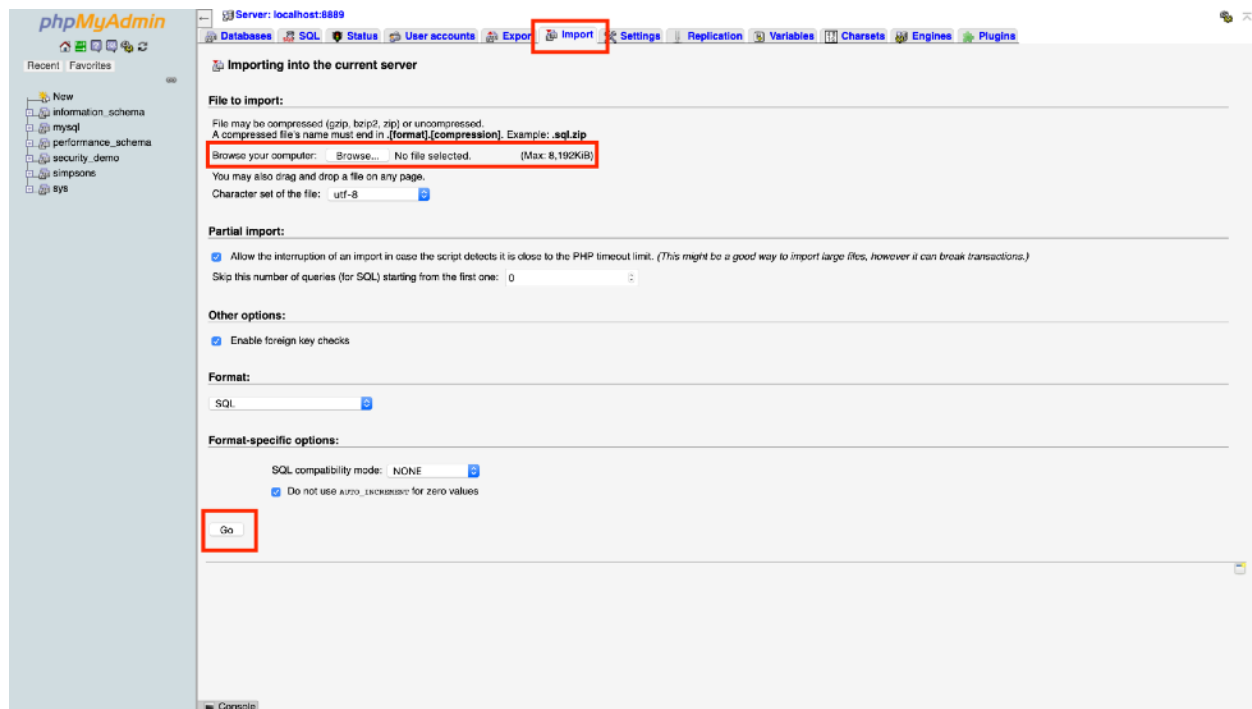


FIGURE 2: IMPORT YOUR WORLD.SQL DATABASE.

That should create the database and load all the information in that file. You will see a list of all the queries it ran and whether or not they are successful.



FIGURE 3: SUCCESSFUL IMPORT OF DATABASE

Now go to your newly created database by clicking on the name "world" in the sidebar on the left of the PHPMYAdmin webpage. You should see 3 tables created, one for cities, countries and languages.

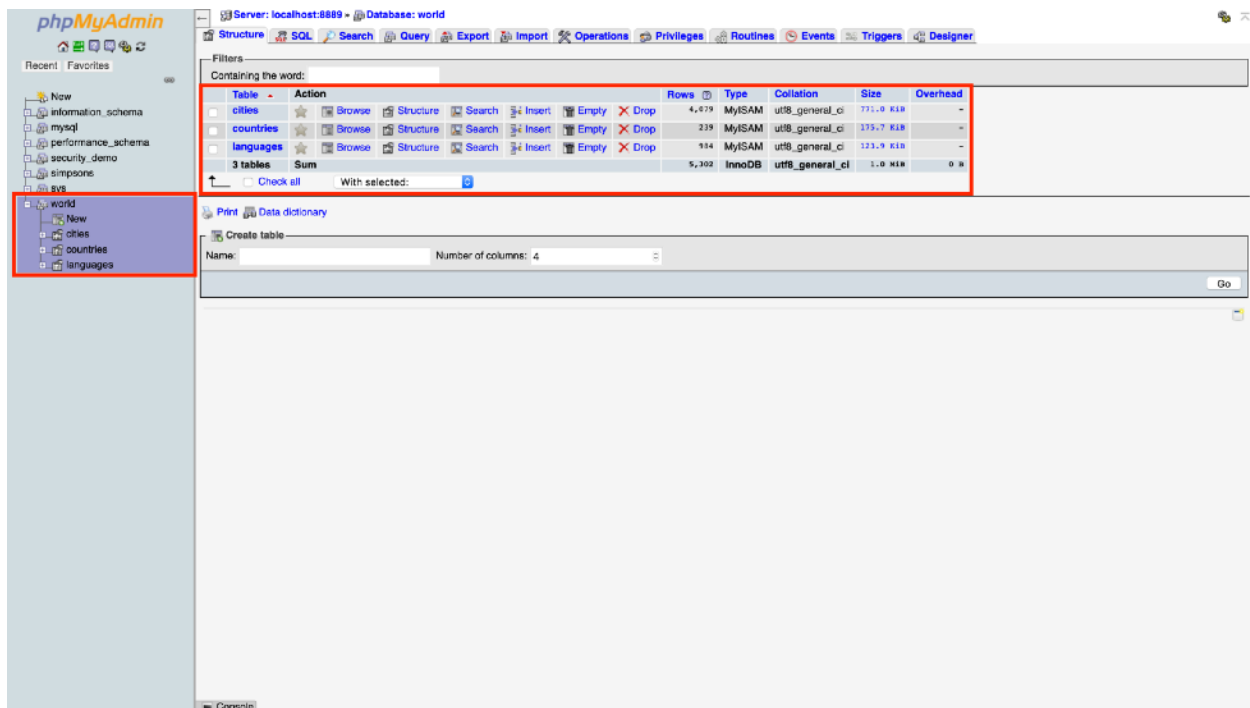


FIGURE 4: CITIES, COUNTRIES AND LANGUAGES TABLES IN THE WORLD DATABASE

Next we will create a MySQL user for your web application to use. To do this, click on the "SQL" tab and then enter the following SQL command, we will use the username **lab5_user**) and the password **password123** for the purposes of this lab.

```
GRANT ALL PRIVILEGES ON world.* TO 'lab5_user'@'localhost'  
IDENTIFIED BY 'password123';
```

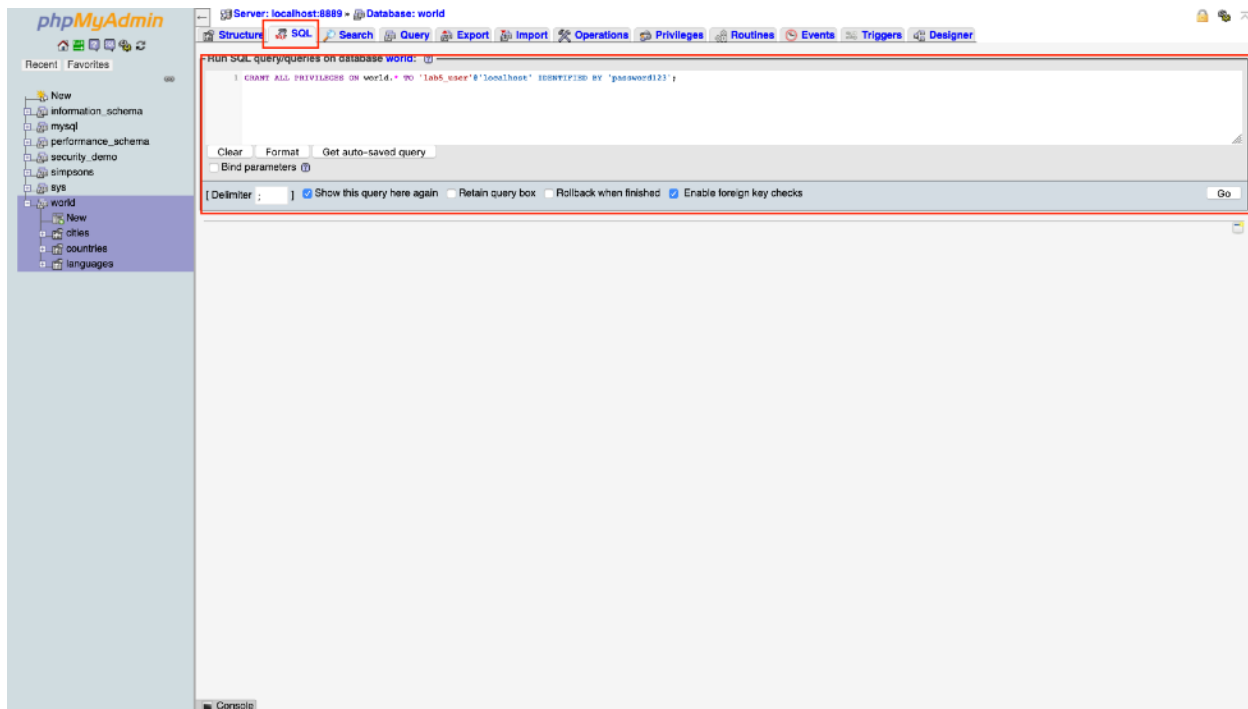


FIGURE 5: CREATE NEW USER AND GRANT PRIVILEGES ON THE WORLD DATABASE.

Then click "Go". That should create the user for the lab and give it access to the world database.

Next, click on the "SQL" tab again and try a few queries to see what information is returned, for example:

- `SELECT * FROM countries;`
- `SELECT * FROM cities WHERE country_code = 'JAM';`
- `SELECT DISTINCT language FROM languages WHERE official='T' ORDER BY language DESC;`
- `SELECT c.id, c.name as city, c.country_code, cs.name as country, c.population FROM cities c join countries cs on c.country_code = cs.code WHERE c.population > 6300000;`

To exit the PHPMyAdmin interface you can simply close the tab in your browser.

Exercise 2 - MySQL in PHP

1. If you haven't already done so, copy the **world.php** file from the starter code to your comp2245-assignment5 folder and ensure you update the **\$username** and **\$password** variables with the username and password you just created for MySQL.
2. In your web browser visit, <http://localhost/comp2245-assignment5/world.php> . You should see a list of countries and the head of state for each.

Exercise 3 - Look up any Country

1. We're providing you the `index.html` file (in the starter code) for a page to do queries against the **world** database. Add it to your comp2245-assignment5 folder if you haven't already done so. Create and Write JavaScript code in a file named **world.js**. You should be able to visit the `index.html` page in your web browser to see a basic HTML page. You may style your page to your liking by creating a CSS file called **world.css**.
2. Modify your **world.php** file to ensure that it will now accept **GET** request variables. You will also need to update your SQL query so it returns information about only the country that is passed to **world.php**. Try visiting your **world.php** in your web browser and appending `'?country=Jamaica'` as the value in the query parameter for the **country** key.
e.g. `world.php?country=Jamaica`

NOTE: You can use the following SQL query:

```
SELECT * FROM countries WHERE name LIKE '%$country%';
```

Remember that you can obtain the value of the GET request variable in PHP using `$_GET['country']`.

3. Now write JavaScript code in your **world.js** file so that when a user clicks "Lookup" button it does the following:
 - i) Listen for clicks on the button with id of lookup. Use the following HTML for the button: `<button id="lookup">Lookup</button>`
 - ii) Fetch the data by opening an Ajax connection to fetch data from `world.php`
 - iii) Print the data you have obtained from the AJAX request into the div with id `"result"`.
4. Since we used a LIKE operator in the WHERE clause of our query, it will also allow us to do partial searches. Try only typing a few letters for a country e.g. "Jam" or "United" or "Pa" and see if the search returns a few extra countries instead of a single country.
5. Next, try submitting no country name and see if it displays all the countries.

Note: Now would be a good time to add and commit your code to your Github repository.

Exercise 4 - Display Additional Countries Information in a Table

Our Country lookup service (`world.php`) currently returns HTML output using a unordered list (``). Now instead of a list, let us change that and have it output some

additional details on the country in an HTML table.

1. Edit your **world.php** file, so that instead of outputting the list, let it instead create and output an HTML **<table>** that prints out the *Country Name*, *Continent*, *Independence Year*, *Head of State* for each country returned by our search. See Figure 6

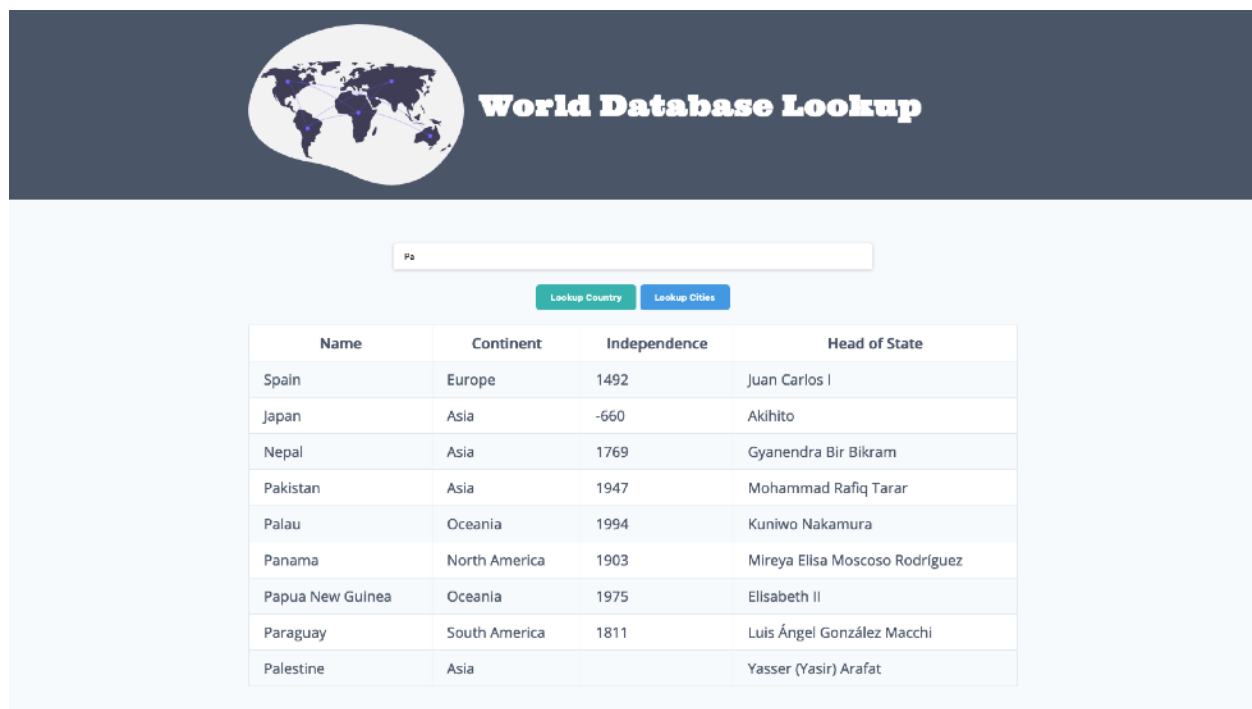


FIGURE 6: COUNTRIES BEING DISPLAYED IN HTML TABLE.

Note: Now would be a good time to add and commit your code to your Github repository.

Exercise 5 - Display Cities in a Country.

Now let us try to get some information on the cities in a particular country.

1. Create a second button in your HTML page with the text "Lookup Cities". This button when clicked will take the country name entered into the search field and return the Cities in that country.

Hint: You may want to ensure that your AJAX request includes an additional query parameter in the query string that helps to indicate that you want to return city information as opposed to the regular country information. e.g. `world.php?country=Jamaica&lookup=cities`

2. You will need to update your `world.php` file so that you check for the lookup query parameter in the query string and if the lookup is set to cities, it runs a different query to return the cities instead of the country information.
3. You will now need to write an SQL query that does an *SQL Join* between the **countries** and **cities** table in your **world** database.
4. And finally you will output the results in an HTML table with columns for *Name*, *District* and *Population* of each city. See Figure 7.



Name	District	Population
Spanish Town	St. Catherine	110379
Kingston	St. Andrew	103962
Portmore	St. Andrew	99799

FIGURE 7: CITIES BEING DISPLAYED FOR THE COUNTRY SEARCHED FOR.

Note: Now would be a good time to add and commit your code to your Github repository.

When you are done and are sure everything is working, switch back to your master (or main) branch and then merge the changes you made in your "php-mysql-implementation" branch back into your master (or main) branch. Then ensure you push those changes back to your Github repository.

Submission

Submit your assignment via the **“Lab 5 Submission”** link on OurVLE, with your Github repository URL. e.g. <https://github.com/yourusername/comp2245-assignment5>