Cardiff University
School of Computer Science & Informatics

OPTIONAL

MySQL

Lab Objective

Use this worksheet to get familiar with fundamentals of creating, using and managing MySQL-databases.

PRELIMINARIES

- This exercise is not assessed - it is optional but could be useful.
- Use examples provided in the lecture slides to help you complete the tasks in this worksheet.
- The School's **MySQL server** is available at `csmysql.cs.cf.ac.uk`. You should use this for all work in this module [1].

  Your MySQLAccount:

  – If you have not set up your MySQLaccount yet, you need to go to `https://dbmanager.cs.cf.ac.uk/mysql/` to create a new account.

  – **Note**: your MySQL account is **not** the same as your normal Cardiff university one, and you should set up a **different password** for this account. You can always reset your MySQL password by going to `https://docs.cs.cf.ac.uk/services/passwords/`.)

  – Further information on MySQL in the School is found at `http://docs.cs.cf.ac.uk/notes/mysql-in-the-school/`.

  – You can create up to 5 databases on your account.

  You can **interact with the MySQL server** in several ways, using:

  - web-based front end phpMyAdmin, available at `https://phpmyadmin.cs.cf.ac.uk`;
  - terminal-based SQL command interface to MySQL (see quick guide: `https://docs.cs.cf.ac.uk/notes/accessing-mysql-from-windows/`);
  - GUI front end called "MySQL Workbench" (see: `https://docs.cs.cf.ac.uk/notes/accessing-mysql-from-windows/`.
  - **For these exercises, we will be using phpMyAdmin.** A short guide on administering MySQL databases using phpMyAdmin is found at `https://docs.cs.cf.ac.uk/notes/administering-mysql-with-phpmyadmin/`

- **NB: VPN connection is required to connect to MySQLand phpMyAdmin.**

(1) Requirements for other modules might differ - always check with your lecturer(s).

EXERCISES

Except where specified otherwise, you **should use** the **'SQL' tab** on **phpMyAdmin** site and **MySQLcommands** and **not** the other options provided by the GUI. This will allow you to get familiar with SQL, which is essential for the web development work you will be doing for the other Flask exercises and assessment.

1. Log in to **phpMyAdmin** using your MySQLcredentials.
2. Go to **'SQL'** tab, and then to **'Console'** which is located at the bottom of the page.

Getting Familiar with COSMC Database Setup

3. Complete the following tasks by executing appropriate SQL queries (*Hint: use the "Reference and Examples" slides from the handout for this session.*):

   3.1. Show all databases associated with your account.

   If your SQL command is correct, you will be presented with a list of available databases.

   Note: you have permissions to modify the database that has the the same name as your username. There is also a database called **`sample`**.

   3.2. Make **`sample`** database 'active', i.e. the database you want to run queries on.

   3.3. Run a command to show all tables in the **`sample`** database.

   3.4. Show the entire contents in the table **`systems`**. You should see output with the following information:

```
+-------+--------+-----------+
| name  | oscode | ownercode |
+-------+--------+-----------+
| blue  |      1 |         2 |
| red   |      2 |         2 |
| green |      3 |         1 |
| brown |      2 |         3 |
+-------+--------+-----------+
```

Working on Personal Databases

For the subsequent exercises you need to use ***your personal database*** - the one that starts with your **user name**! Do not use the **`sample`** database, as you do not have write permissions for it.

4. Create a new database **`cmt120_practice`** by going to https://dbmanager.cs.cf.ac.uk/mysql/(2)
5. Go back to **phpMyAdmin** and refresh the databases by clicking on the green circular arrow.
6. Go to **'SQL'** tab and, using an appropriate SQL command, switch to this database (i.e. select it to work on it).
7. Check if you have any tables (there shouldn't be any yet).
8. Create a table called **`staff`**, similar to the one in the lecture notes:

```
CREATE TABLE staff (staffNo varchar(10), firstName varchar(30),
lastName varchar(30), position varchar(20),
salary int, branchNo varchar(10));
```

Verify that the table has been created (albeit empty at the moment), examine the table's structure (*Hint*: **`describe`**), and display the table's content.

Notes and further comments:

When creating a table, you need to consider which data types the attributes should have. For instance, in the above table we chose to store first names as strings of variable length up to 30 characters (i.e. **`varchar(30)`**), and salaries as integer numbers (i.e. **`int`**).

In addition to the data types already mentioned in the lecture, explore http://www.w3schools.com/sql/sql_datatypes_general.asp to find out more about the available data types.

(2) This can also be accessed through https://docs.cs.cf.ac.uk/, then 'Password Management' and 'MySQL Password'.

Further, when creating a table you may want to specify additional options and constraints for your attributes - see the next section for details (and in particular, the exercises on **ALTER TABLE**). Usage examples include:

- To make sure that the **firstName** attribute always has a value (i.e. to disallow **NULL** values).
- To set the default value for salaries to be 10,000.

Use the documentation at http://www.w3schools.com/sql/sql_constraints.asp to find out more information on constraints.

9. Delete the table **staff**, and check the table has indeed been deleted. Then create the table again.

Primary Key and Altering Tables

10. You should always define a **primary key** for your tables. In some cases, one of the existing attributes of the data that you are modelling may serve as the natural primary key (for example, unique student number), in other cases you may need to create a surrogate key just for the purpose of uniquely identifying the rows. Primary key may even consist of multiple attributes. Read http://www.w3schools.com/sql/sql_primarykey.asp to find out about the syntax for the primary key constraint.

    For the sake of example, let's use **staffNo** as the primary key in our table **staff**. We can declare it by using the following command:

    ```
    ALTER TABLE staff ADD PRIMARY KEY (staffNo);
    ```

    (Or alternatively (and normally) this is done when you initially create your tables.)

11. Inspect (**describe**) the table's structure. What has changed?

12. You can use the **ALTER** command to alter other properties of your tables after they have been created. For example, to make sure that the salaries are never unspecified and the default value is 10,000:

    ```
    ALTER TABLE staff MODIFY salary INT NOT NULL DEFAULT 10000;
    ```

    Inspect the table's structure again. What has changed?

13. Alter the table again to make sure that the **firstName** attribute always has a value.

    Inspect the table's structure one more time. What has changed?

Data Import and Export

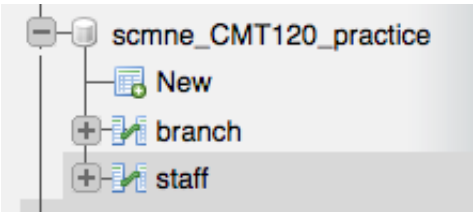
14. Let's populate the table with some data.

    14.1. Using **INSERT** command, insert a record into **staff** table for John White:

staffNo	firstName	lastName	position	salary	branchNo
SL21	John	White	Manager	40000	B005

    14.2. Data can also be imported from a file. To try this, download the file **staff.csv**(3) to your computer:

    - Ensure you have selected the table you want to import into, by clicking on the table name on the left-hand side menu. The selected table will be highlighted, as shown in the picture below. (If you do not select the table you want to import data to, **phpMyAdmin**will create a new table for you, which will be named after the file name and without the attributes we need.)



    - Go to **'Import'** tab on **phpMyAdmin**interface, browse your computer to choose the downloaded file, and then, leaving all default values as they are, click on **'Go'** button (you might need to scroll to the bottom of that frame.)

    - Run an appropriate MySQLcommand to verify that the import worked, i.e. display all data in the table **staff**. You should see the same data as in the lecture notes.

15. Similarly, using SQL commands, create a table called **branch** using the appropriate data types for the attributes. Do not forget to define a primary key using appropriate data item!

16. Populate this table from CSV file **branch.csv** (also available for downloading).

(3) This is simply a comma-separated (CSV) human readable table, containing the data seen in the lecture notes.

17. Check the table data is the same as in the lecture slides, i.e.:

branchNo	street	city
B005	22 Deer Rd	London
B007	16 Argyll St	Aberdeen
B003	163 Main St	Glasgow
B004	32 Manse Rd	Bristol
B002	56 Clover Dr	London

18. Write an SQL command to add a record for another branch to table **branch**, with number **B008** located at **12 Millbank, London**. Examine your table to check the new branch's details have been added to the table.

19. Assuming **staffNo** was declared as primary key in the **staff** table, what would happen if you tried to add another record with the value of **staffNo** that already existed in the table?

    For example, try the the following query (check and confirm that **SL21** already exists before you run the query):

```
INSERT INTO staff VALUES
("SL21", "Horatio", "Nelson", "Vice Admiral", 1337, "B001");
```

    What is the result of the above query?

20. Write an SQL command to modify[4] the address of the branch **B008** to be *PO Box 3255, London*.

21. **phpMyAdmin** can take take of data **export** for us, using various file formats, e.g. CSV, SQL, JASON files. Go to the **'Export'** tab to explore options by trying exporting a table as one of the file types of your choice.

(4) See http://www.w3schools.com/sql/sql_update.asp

Further Practice on Writing SQL Queries

If you are happy with *essential* tasks 1 - 21, you can stop here. If you would like to stretch yourself a bit further, complete the tasks in this section. Possible solutions are available.

Write SQL queries to answer the following questions (or carry out operations). Use examples provided in the lecture notes to help you define the queries.

There can be multiple possible solutions to some of the questions.

1. In which branch does David Ford work?
2. Which employees work in branch **B003**?
3. Which branches are located in London?
4. What are the names of employees who have salary greater than $20,000$?
5. What is the average salary? The largest? The smallest?
6. What are the names of employees who have a below average salary? Above average?

   (Hint: Use a sub-query to find the average salary first, then filter the employees using the result of this sub-query.)
7. Produce the list of all employees sorted by salary in ascending order. Do the same in descending order.
8. Who are the three lowest paid employees?

   (Hint: use data selection limit , e.g. see: `http://www.w3schools.com/php/php_mysql_select_limit.asp` *.)*
9. Determine the set of all job titles (positions). That is, find the list of all job titles without duplication.

   (Hint: `http://www.w3schools.com/sql/sql_distinct.asp` *.)*
10. What are the names and salaries of employees who work in London?

    (Hint: since the information about names and salaries is in table ***staff*** *while the information about cities is in table* ***branch****, you need to do a* join.
    See `http://www.w3schools.com/sql/sql_join.asp` *.)*
11. Are there any assistants working in Aberdeen?
12. Employees in which city have the highest salary?
13. What is the top salary in Glasgow?
14. Which branches do not have a manager?

15. Which branches do not have any employees?

16. Find all last names ending with an *"e"*.

17. Reward Susan Brand who works in B003 as a manager by raising her salary to $36,000$.

18. In fact, let's reward all managers: raise their salaries by 10%.

19. Branch **B007** is closing. Remove all employees working there, and remove the branch itself.

20. Hire a new employee. Her name is Mary Smith and she will be working as an assistant in the Glasgow branch with the starting salary of $22,000$.

Useful addresses	
COMSC MySQLdatabase server: COMSC **phpMyAdmin**:	**csmysql.cs.cf.ac.uk** https://www.cs.cf.ac.uk/phpMyAdmin
MySQL manual: SQL statement/syntax index:	https://dev.mysql.com/doc/refman/8.0/en/ * https://dev.mysql.com/doc/refman/8.0/en/dynindex-statement.html * * You can select documentation for other versions as well on that page.
SQL tutorial and reference:	http://www.w3schools.com/sql/
SQL and MySQL cheat sheets *(in no particular order)*	**[1]**, **[2]**