



## Assessed Coursework

|  |                  |                                   |  |
|--|------------------|-----------------------------------|--|
| Course Name                                    | DSR              |                                   |  |
| Coursework Number                              | 2                |                                   |  |
| Deadline                                       | Time:            | 17:00                             | Date: Friday 15 <sup>th</sup> November |
| % Contribution to final course mark            | 20%              | This should take this many hours: |  |
| Solo or Group ✓                                | Solo ✓           | Group                             |  |
| Submission Instructions                        | Submit on Moodle |                                   |  |
| Who Will Mark This? ✓                          | Lecturer ✓       | Tutor                             | Other                                  |
| Feedback Type? ✓                               | Written ✓        | Oral                              | Both                                   |
| Individual or Generic? ✓                       | Generic ✓        | Individual                        | Both                                   |
| Other Feedback Notes                           |                  |                                   |  |
| Discussion in Class? ✓                         | Yes ✓            | No                                |  |
| Please Note: This Coursework cannot be Re-Done |                  |                                   |  |

### Code of Assessment Rules for Coursework Submission

Deadlines for the submission of coursework which is to be formally assessed will be published in course documentation, and work which is submitted later than the deadline will be subject to penalty as set out below. The primary grade and secondary band awarded for coursework which is submitted after the published deadline will be calculated as follows:

- (i) in respect of work submitted not more than five working days after the deadline
  - a. the work will be assessed in the usual way;
  - b. the primary grade and secondary band so determined will then be reduced by two secondary bands for each working day (or part of a working day) the work was submitted late.
- (ii) work submitted more than five working days after the deadline will be awarded Grade H.

Penalties for late submission of coursework will not be imposed if good cause is established for the late submission. You should submit documents supporting good cause via MyCampus.

**Penalty for non-adherence to Submission Instructions is 2 bands**

### Marking Criteria

(detailed in the Assessment section below)

**Data Storage and Retrieval**  
**Course Work 2**  
**Wednesday 6<sup>th</sup> November 2024**  
**Database Creation and Querying**

**Coursework Exercise 2:** In this coursework, you will create a database based on a model answer ER diagram for a scenario of the same structure as coursework 1, and then perform some queries on your database. There are four tasks to complete, listed below, including writing a report that should be submitted via Moodle.

**Task 1:** Starting from the coursework 2 model ER diagram provided on Moodle, create a relational database. To do this you should first convert the ER Diagram to a corresponding relational schema. For this task, you need only define the set of tables you need for the base relations and start relating them. Do not worry about populating or querying the database until this task is done.

**Note:** The text below assumes that you are using MySQL workbench to connect to a MySQL database server. However, you can use any Database Management System that you prefer, as long as you are able to **include screenshots in the report that clearly show your queries being run and your query results being printed out in a tabular format** and you are also able to **include a screenshot showing pictorially the structure of your constructed database e.g., an EER Diagram.**

You should follow these steps:

1. Firstly, examine the model ER diagrams for coursework 2.
  2. Using the process presented in the lectures, convert the model ER diagram into a set of relations. You should work on the relations and schema on paper first.
  3. Only once you have your schema written out on paper, can you start to input them as **tables** into MySQL Workbench. Choose carefully the appropriate datatypes for each attribute.
  4. Make sure you follow the rules for adding foreign keys (FKs) depending what the relationship is in your ER Model (1-1, 1-N and N-M).
  5. In the “reverse engineer schema” view – make sure the foreign key relationships between the tables are represented in your database.
- Check that you have all the tables you need in your database.
  - Check that you have assigned appropriate keys to all your relations.
  - Check that you have created all of the appropriate foreign key relationships.

**Task 2:** Populate your database with the supplied data. The data is available as individual .csv files (and pdf file) in the Coursework 2: Database Queries Report section of the Moodle page.

You can enter the data into your database in one of two ways:

- (1) Import the data as text files into the table using the import facility (see 'Importing Data From Files').
- (2) Enter the data manually row by row in the Results grid view (see 'Importing Manually')

**NB** – before you do either of these – you must check the following:

1. Your tables have the same number and type of attributes/fields as the tables in the text files/on the sheet
2. The attribute/field names are in the same order
3. The data types of the actual data and those that are in your tables match.

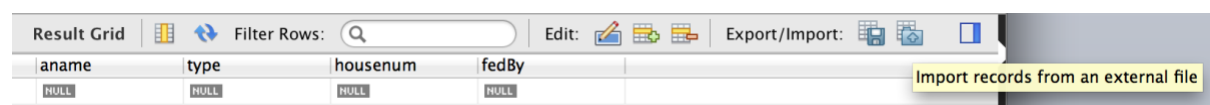
If any of the above does not match, then you can change your database **OR** the order of data in the text files. However, note that if you change the data then your queries might not produce the expected results, so **it is best to avoid changing the data if at all possible**.

The order that you populate the tables is important: you should populate the entities **BEFORE** populating the weak entities or relationships, to ensure that foreign key constraints can be met – remember, the value inserted in a foreign key column **MUST** already exist in the referenced primary key. Plan ahead the order that you are going to import data into the tables.

### Importing Data from Files

The data is also provided as .csv files.

- (i) Download the data (text files).
- (ii) Check that the data in the text files matches the column headings in your tables. You can use Notepad++ (or similar) to open the files – using Excel is *not* recommended!
- (iii) If data in text files doesn't match your tables – as above, you can change the text files, but updating your database schema is preferable.
- (iv) Import the text files one by one into all the tables in your DB. Select the '**Import Records**' button from the Results grid toolbar (see Figure 2).



**Figure 2: Importing external data (text files) into your database tables in MySQL Workbench.**

Select the file you want to import (they should all be saved locally) and press Open. The Results grid is now updated – this allows you to check that the data is in the form that the table expects. You **MUST** now click the Apply changes to upload the data to the server. Only at this stage will you be informed of any problems importing the data (see the next section).

## Dealing with Importing Problems

We expect you to have difficulties in importing the data. Dealing with such problems is typical of working with data in real-life databases and learning to identify what the problems are is important. Problems when importing are usually caused by one of the following:

1. *data type mismatch* – “ERROR 1406 (22001): Data too long for column 'xxx' at row 1”. This happens if declared data types and the type of the data in the text files don't match. Change the types to match by altering the table definition. The ordering of columns being different between the table definitions and the data files might cause this.
2. *data type mismatch for NULL* – “ERROR 1366: 1366: Incorrect integer value: 'NULL' for column xxx' at row 1”. This happens if you are trying to import a NULL value as a String, in quotes. You can edit the SQL for the insert manually to name 'NULL' into NULL.
3. *primary key constraints* – “Error Code: 2627. Violation of PRIMARY KEY constraint. Cannot insert duplicate key in object”. This happens, for example your primary key includes too many or too few columns. Check that your table definition has an appropriate primary key.
4. *foreign key constraints* – “Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails”. This happens, for example, if you are importing the data for a foreign key, but haven't yet populated the referenced primary key yet. *Think carefully about how to import values into Staff table.*

**NB** - Before moving on to the queries - check all your data has been imported for all tables by viewing each table in the Results grid. The important outcome is to populate your database with the data from the example text files for you to be able to query the database properly.

## Importing Manually

The Results grid offers a way to manually enter data. To use this, right click the table you wish to populate in the left hand side Schema explorer, then click 'Select Rows – Limit 1000'. You can then enter the data for a new row by clicking the last row (the one full of NULL values) – see Figure 1. After you enter a new row, click Apply to send that row to the database.

| aname   | type  | housenum | fedBy |
|---------|-------|----------|-------|
| Dusty   | sname | NULL     | NULL  |
| Fluffy  | dog   | 34       | Jim   |
| Red     | dog   | 42       | Jim   |
| Robin   | dog   | 42       | Jenny |
| Splodge | cat   | 34       | Jim   |
| Tinky   | dog   | 38       | Jim   |
| Spot    | dog   | NULL     | NULL  |
| NULL    | NULL  | NULL     | NULL  |

Animal 1\*

Apply Revert

**Figure 1: Manual data entry in Result grid view in MySQL Workbench**

**Task 3:** Perform some queries on the data and then **save the results**.

Given the small amount of data in your database, it would be possible to answer the following queries by just scanning the tables. However, the point of this exercise is to become familiar with the techniques for querying, by practicing identifying the appropriate SQL design pattern, and creating the correct SQL to query the database.

- As you perform your queries, **save the results** using the following naming scheme: Query <ID>. For example, the first query will be saved as 'Query 1'.

### An Example


You are tasked to *“Find the name of the student with Student ID 1235000”*.

**Step 1: Consider the possible SQL Design Patterns.** In this example, all of the data we need is within a single row of the Student table, so the Basic query pattern is appropriate.

**Step 2: Identify what is needed to apply the pattern: projection(s), table(s), condition(s).** We need to project forename and lastname attributes from the Student table. The condition we need to apply is selecting only row(s) where the matric\_no column must be equal to 1235000.

**Step 3: Generate the new SQL query from the design pattern and what you identified in Step 2.**

```
SELECT forename, lastname
FROM Student
WHERE matric_no = 1235000
```

**Step 4: Enter the query into MySQL Workbench.** Make a new Query Tab in MySQL Workbench (File...New Query Tab), and enter your SQL query. Click the Execute button  to run the query. If you have a *syntax error* (such as misspelling an attribute name) in your SQL, examine the Action Output at the bottom of the screen, then correct the error. Otherwise, the results will appear in the Results grid (see Figure 3 below).

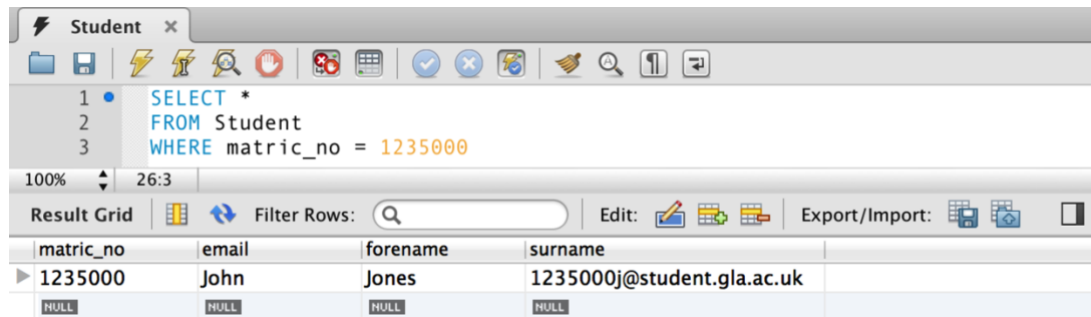



Figure 3: Viewing results of a query in MySQL Workbench.

**Step 4. Close the Query Tab, export and naming the SQL and results of the query.** Use the Export button  on the Results grid to save your results in CSV format. Finally, use the File...Save Script As to save the SQL query you ran as a file with a .sql extension.

#### Queries to Run:

Write queries, and save the results and SQL queries. For queries 1-4, **also write out the relation algebra**. Remember to consider the SQL design patterns for each query.

1. Find the job title (i.e., lecturer, professor, etc.) of staff member Stephen Brewster. [1]
2. Find the exam results for all students apprentices taking CS-1Q. [1]
3. Show the results of query 2 with the students' apprentices' names, and order the results in ascending order alphabetically by surname. [1]
4. Find the names of all staff lecturing on 'CS-1Q' who are also professors. [1]
5. Find the number of staff members of each job title (Hint: consider the Grouping design pattern). [1]
6. Find the names of tutors who are not lecturers and whose tutorial group meets in room 11. (Hint: (1) Create a query to select the tutors whose group meets in room 11 and then (2) extend the first query using the join pattern with the Staff table.) [1]
7. For each assessed student, display their student ID and their average exam and practical marks. [1]
8. Show the results of query 7 with the students' names, but only for those with the surname 'Smith' or 'Saunders'. [1]
9. Find the names of staff who manage themselves. [1]

**Task 4:** Produce a report (to be submitted in PDF format) detailing your database design, your queries and some reflections about how you built the database (including a critical reflection on your choice of DBMS, e.g. pros/cons, ease of use, documentation/tutorials etc).

Your report should include:

**1. Your database design– (12 marks)**

Provide the EER diagram (or similar) **of your constructed database**. (you can take a screenshot of the diagram in MySQL Workbench).

**2. Queries & Results - (9 marks)**

Provide the SQL and results for queries 1-9. You can copy in the text of the query, and the CSV or take a screenshot of the relevant **part** of MySQL Workbench. For each query, state what SQL design pattern you used for the query.

For queries 1-4 ONLY, also provide the relational algebra [don't worry about the ordering for query 3 for the relational algebra].

**3. Challenges & Reflections - no more than 400 words (3 marks)**

(i) Summarise the main design and building process for constructing a database. Discuss why are ER diagram necessary in designing the database? And, why is the transition from conceptual to logical design important?

(ii) Discuss the challenges you faced when constructing your database. What major problems did you have in carrying out the building and querying of the database, and how did you overcome them? Also, include a reflection on your choice of DBMS.

(ii) Include with a brief reflection on your personal achievements in doing this work (in other words, what you now know that you didn't know before or what you can now do that you couldn't do before). Consider your achievements in terms of their importance and the level of effort you had to expend. If you still have particular difficulties (i.e. things you believe you should have achieved, but haven't (yet!)), identify them. This will be useful to you when revising.

**Assessment:**

This exercise is worth 20% of your overall course grade. The exercise is marked out of 24. A total of 30 marks are available for the two pieces of coursework for the course.

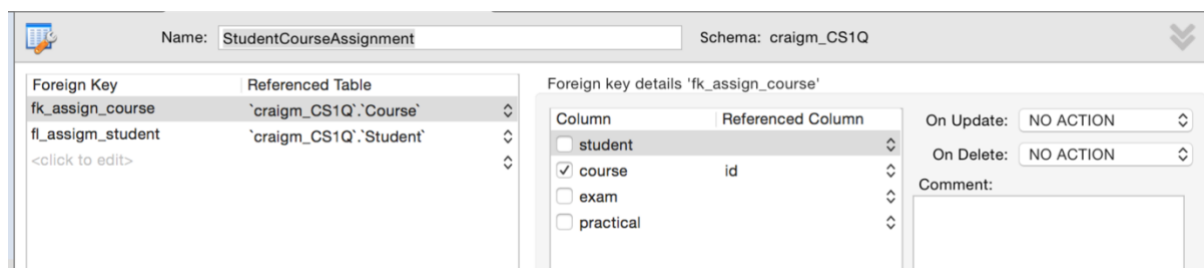
| <i>Assessment</i>     | <i>Marks</i> |
|-----------------------|--------------|
| Database construction | 12           |
| Queries & RA          | 9            |
| Report quality        | 3            |

Your report should be submitted electronically via Moodle – there will be a submission link in the Coursework 2 section of the course Moodle page. Make sure your name and student ID number is clearly visible **in the submitted document** (e.g., naming your file as your student ID is not sufficient and marks will be lost if your details are not on your submission).

Submissions are due before 5 pm on **Friday 15<sup>th</sup> November 2024**.

## MySQL Workbench Gotchas:

- Workbench has strange error messages if you start entering foreign key constraints before the table has been created – you need to have “Applied” the creation of the table before starting to create the foreign keys.
- Foreign keys are named constraints, and hence need globally unique names. Its recommended that foreign key names mention both the source and target table names, e.g. “fk\_assign\_student”.



- “Error Code: 1215. Cannot add foreign key constraint” probably means the foreign key creation is between columns of different types (e.g. INT vs VARCHAR), or that the referent column was not a primary key.
- Workbench can get confused with too many open Table Editor tabs. Be systematic in opening and then closing tabs. When things go wrong, close all tabs.
- Apply changes as often as possible, e.g. when deleting a column, and then remaking it with the same name but a different datatype, needs an Apply.
- Reordering columns in a table can be problematic if those columns are involved in a primary key.