Assignment One: Database Design and Implementation

Weight 20%, marks out of 60 **Deadline: 15:30 (GMT) 28 February 2017**

Overview

The coursework takes the form of a group project in preassigned groups. The first section of the coursework involves the design and implementation of a relational database (using MySQL). To get a good mark, your coursework must demonstrate many of the advanced topics covered in lectures rather than merely a repeat of material covered in the first year database course.

Groupwork

Share the work among the group members, ensuring that everyone has an equal share. If you work together at the same machine, swap the 'driving position' regularly (every 10 minutes?) to ensure that everyone contributes.

Experience gained in these topics will help in the examination. Some group members could work on a section, and then the others could review it and make suggestions for improvements – that way everyone knows about what is going on at each stage.

If necessary, marks will be adjusted if it is obvious that some students have not participated enough. **Please provide a summary in your report stating the contributions made by each group member.**

Collaboration and Plagiarism

Coursework reports and code must be the group's own work. If some text or code in the coursework has been taken from other sources, these sources must be properly referenced. Failure to reference work that has been obtained from other sources or to copy the words and/or code of another student is plagiarism[1] and if detected, this will be reported to the School's Discipline Committee. If a group is found guilty of plagiarism, the penalty could involve voiding the course.

Students must never give hard or soft copies of their coursework reports or code to students in another group. Students must always refuse any request from another student not in their group for a copy of their report and/or code. It is expected that all group members will have read and write access to the report and code for their group.

Sharing a coursework report and/or code with another group is collusion, and if detected, this will be reported to the School's Discipline Committee. If found guilty of collusion, the penalty could involve voiding the course.

[1] Heriot-Watt guidelines on plagiarism https://www.hw.ac.uk/students/studies/examinations/plagiarism.htm

The task

As you proceed through the project, you might need to change some earlier decisions. You should:

Task	Description	Marks
1.	Extend the scenario for your application and draw an ER-diagram that captures these requirements	10
2.	Translate the ER-diagram into a relational schema	10
3.	Implement the schema in MySQL	5
4.	Fill the MySQL database with data	5
5.	Define roles, access permissions, and views for the users of your database	5
6.	Write a set of queries for your MySQL database	10
7.	Define suitable indexes for your tables	5
8.	Add details of your tables, views and queries to the supplied java application	10

Task Summary and Timeline

When	What
Weeks 1 – 4	Get T2 ER diagram and RM tables checked
Week 6	Aim to have T1-T7 finished
Week 8	Submission, demo T8 JDBC app in lab

Submission

- Report through TurnItIn assignment on Vision
- Archive file as a zip or tar.gz (no rar) file containing all scripts and source code through GradeCenter assignment on Vision.

T1 Scenario and Conceptual model (10 marks)

"A sports centre needs a booking system to manage members and allow them to book rooms and equipment."

Design a database according to the fairly general request above. You need to extend the scenario with more details as appropriate to fit the requirements of this database exercise – for example, take into account the features that you have to include in the ER diagram (see below), the fact that you need a few different roles of users (e.g. staff and members), and the type of SQL query that you are asked for. You may consider the type of rooms that are available (e.g. climbing wall) and whether members need to have passed certain training courses to use the equipment.

You are NOT asked to produce a complete working application or a complete set of queries that the company would actually need to run your database system successfully. Rather, you are asked to use this scenario to demonstrate your proficiency in the various aspects of this assignment. You will need to extend the scenario quite a bit, and invent some extra details.

Write a description of your extended scenario, so that the reader can understand what your diagram represents.

Create an ER diagram for the conceptual model of your scenario. The diagram MUST include one-to-many and many-to-many relationships, ideally with a variety of participation constraints, and several other advanced ER features such as a recursive relationship; derived attributes; composite attributes; composite keys; repeating attributes; use of generalization and specialisation. The diagram should contain a minimum of 9/12/15 (entities and relationships) for a group of 3/4/5 students, e.g. Employee Works for Department and Department has Employee as manager => 2 entities and 2 relationships = 4 (entities and relationships).

This ER diagram should **not** show foreign keys (this comes in T3), it should only show the links between entities as relationships (linking lines). Each attribute only occurs ONCE in the diagram. You might need to use a drawing tool like yEd[2], although Microsoft Visio is another option; rather than a relational database tool which will put in the foreign keys.

Make a note of any details of the requirements that could not be captured by the diagram.

Get this ER diagram checked in tutorials before continuing.

Deliverable: Report to include the extended description of the scenario, the ER diagram for your database, and any supporting notes for you diagram.

T2 Translation into Relational Schema (10 marks)

Define a relational schema from your conceptual ER diagram. Use the steps given in the lectures to convert your conceptual design into a relational schema. The schema will include foreign keys, and extra tables where required, e.g. for many-to-many relationships. Primary and foreign keys should be clearly identified. Make a note explaining how it has been derived from the original conceptual ER diagram, for those cases where you have had to make a decision.

Create a data dictionary for your tables – for each table, include column names, domain and other constraints (including keys).

Check that your tables are all in 3rd Normal form (i.e. no columns just dependent on one part of a composite primary key, and no columns dependent on a column which is not a key). Make changes to your tables if you decide it is necessary.

Deliverable: Report to include schema as a data dictionary clearly identifying data types, and primary and foreign keys. Add notes for any unusual steps that were followed.

T3 Implementation of the Schema in MySQL (5 marks)

Create the tables either using phpMyAdmin or through the SQL command line, using the innoDB storage engine.

To use the command line, you will need an SQL script containing `CREATE TABLE` and `ALTER TABLE` commands. If you use phpMyAdmin, you can run the SQL script, or create the tables using a GUI and finally export this SQL script (and check it makes sense, e.g. not multiple identical indexes).

[2] https://www.yworks.com/products/yed accessed December 2016

The data definition language statements must specify:

- appropriate types for the attributes
- the primary key;
- constraints such as NOT NULL and UNIQUE whenever appropriate;
- default values if appropriate;
- FOREIGN KEY constraints, together with the policy for reacting to changes
- Write comments into the script that explain the rationale behind the definition of your constraints.

There is a document on Vision on how to use MySQL in the linux lab or using phpMyAdmin.

Include your commented SQL script containing the data definition language commands in your report.

Deliverable: Commented SQL script should be included in your archive file.

T4 Loading Data (5 marks)

Load interesting data into the database, experimenting with insert statements and the bulk loader. The data should be realistic, and there should be enough to enable you to demonstrate a range of queries in T6.

Write a report stating which data was loaded from insert statements, which from the bulk loader, plus the contents of a few sample related records from each table – this is to give the marker a feel for whether you have entered suitable data, without having to read insert statements or poorly formatted output from SELECT *.

Deliverables:

- Report to include a description of the data loading followed.
- Archive file to contain the data loading scripts in a loader directory.

T5 Roles, permissions and views (5 marks)

Define some typical users and roles. (You don't have permission on the departmental MySQL server to actually define users and roles, so you don't have to implement this.)

Define and implement several views to give restricted access to a subset of the data. Produce a grid showing which users can access which tables and views.

Create a report explaining your typical users, and for each view provide a description in English, the reason for creating this view, and the SQL script used to create the view. Include the access grid in the report.

Deliverables:

- Report to contain a description of each role and view created.
- Archive file to include script to create the views.

T6 Queries over the Database (10 marks)

Write a set of transactions for your scenario, some of which must contain multiple CRUD (Create, Read, Update, Delete) operations. Create one or more views, for example to simplify queries with a frequently used base query.

Groups of 3/4/5 should write at least 9/12/15 essentially different transactions.

Many queries are likely to contain search condition(s) in which the data is supplied at run-time. However, for the purposes of T6, you should supply the query with data, e.g. `WHERE gender='M'`.

Among your set of queries the majority should be interesting in the sense that they contain at least one of the following features:

- joins involving a composite key,
- joins involving the same table twice,
- aggregation with group by and having,
- nesting with aggregation,
- nested negation, involving NOT EXISTS or NOT IN
- using MySQL built in functions
- using views

Don't provide a large set of basic queries, since you will not get many marks for this.

Deliverables:

- Report to contain a description for each transaction in English, the SQL, and the output or a subset of the output if there is a lot of it.
- Archive to contain a script file for each transaction. These script files should be included within a queries directory in your archive file.

T7 Indexes (5 marks)

Define appropriate index definitions that are designed to speed up the queries that you have created. Write an SQL script containing a set of `CREATE INDEX` commands. A group of 3/4/5 should define at least 3/4/5 indexes.

Deliverables:

- Report to contain an explanation why each index is needed. If your design doesn't need at least 3 of your own indexes, explain why not.
- Archive to contain a script containing the index commands

T8 Java Application (10 marks)

For this part of the coursework you should extend the provided Java application to connect to your database. Download the Java application from Vision and refer to the separate document explaining the Java application.

Alter the class JDBCMySQLApp to provide a selection of your own queries to run in the application. A sample set of queries is already there, and these should be altered or deleted. Instructions on what to alter is in the code. Groups of 3/4/5 should ideally supply 6/8/10 queries. Your queries could be a subset of the queries in T6, or some other suitable queries. Particularly, ensure that you include some prepared queries, updates of some kind, and some which are based on user input. It is more important that you demonstrate the range of types of Java coding rather than the range of SQL queries. Don't forget to alter my comments if necessary. If allowing the user to type in data, do some form of validation.

Deliverables:

- Archive to include your JDBCMySQLApp source code in an app directory.
- Demo your application during a lab session.