IS 503 – Fall 2022/2023
Term project
Deadline: 20.01.2023 (no late submission)

For your term project, you will design a database, document your design, and prepare queries that would be used by the application for your case. A user interface is not needed, think as if you are only responsible for the database layer of an application. The aim of this project is to learn how to design an application that utilizes a database. You can work on this project on your own or in a group of a maximum of three students (another two student and yourself). Please inform us here about your group (even if you are on your own) until 07.12.2022.

Your project case is given below, but the information given related to the project case is merely for inspirational use. You are free to enrich and substantially change the properties of the project ideas, given that you are able to come up with deliverables that are qualified and complex enough to satisfy the project requirements.

Project case 1: CoderSpace

You are required to design a skill-based company talent matching tool like Coderspace. Participation in the application can be both as a company and as a programmer. Users who join as programmers create their resumes by entering their experience information and solving the aptitude tests in the application. Users who are registered as a recruiter can set the recruitment criteria for the positions they interest, and determine the success criteria for the tests on the application. In addition, companies can organize challenges for programmers to recruit. For other features of the application, you can get an idea by visiting the application web page.

Project requirements

- Create your project schema using your METU mail address. For example, if your address is "john.doe@metu.edu.tr," name your schema as "john_doe" (using period is relatively problematic, you can simply use an underscore). If you have a teammate, you can choose one of the two or combine them. Your schema design must make sense in detail. **(20 pts)** For example, while it is technically possible to store a date in a TEXT column, it would not make much sense.
- Your schema must have at least six tables **(5 pts)**, 20 columns **(5 pts)**, and 75 tuples (rows) **(2 pts)** in total.
- Your schema must have at least one table having a composite key **(5 pts)**.
- Your schema must have at least one TEXT column **(3 pts)** and at least one DATE/DATETIME/TIMESTAMP column **(3 pts)**.
- All necessary foreign key relationships must be set **(10 pts)**. You must specify the referential integrity actions for insert/delete/update operations while creating the tables.
- For each group member, two complex queries with **at least two joins** and **a correlated subquery (i.e. if there are 2 students in the project team, there have to be four complex queries and two correlated subqueries.) (10 pts)**:

  - At least one of them must use partial matching using LIKE.
  - At least one of them must use EXISTS or NOT EXISTS.
  - If you have a teammate, both of your queries must satisfy these requirements. For example, you need to have at least two queries with EXISTS/NOT EXISTS in total.
  - These queries must make sense such that they must correspond to actual possible use cases of your application. Put them and their explanations in the separate SQL file that is provided.
- Create an analytics-focused view that could be used to make some aggregations for the dashboard of the application. Include it in your schema export and your query file. **(5 pts)**
- Create two triggers for analytical purposes. For example, these triggers could update a counter column in your tables and calculate a sum or average (or another descriptive statistic). Think about the derived attributes of your first assignment. Include them in your schema export and your query file. At least one trigger must use an aggregate function. Note that you will also need to show that your trigger works by providing some control queries. **(12 pts)** For more information, check the query file.
- Export your schema as a single .sql file and include everything like the triggers and the create statement in the export file **(5 pts)**. You can check the MySQL Workbench guide to learn more. Note that the omission of your export would also cause you to lose points from the other related criteria.
- Prepare a project report. Explain your project's case, scope, capabilities, and assumptions. Basically, someone who has no idea about your design must be able to fully grasp your design by reading the report **(5 pts)**.
- Your report must have a corresponding **ER diagram (just like the ones you saw in class and drew for the first assignment)** in your report **(10 pts)**. You cannot simply create a diagram from your tables using MySQL's feature.

Deliverables

- Your project report
- Your schema export
- Your query file (fill in the .sql file that comes with this project brief)

Compress your project report, query file, and your schema export together and submit it. Only one member of the group should submit it to prevent conflicts. Please include the group members' names in the project report.

Tips:

- Use the requirements list as a checklist before submission.
- Make sure your queries work.
- Do not delay the implementation. You will possibly have some problems that can cause you to redesign your schema.
- MySQL Workbench has some bugs. For example, if it does not let you create a foreign key constraint, you need to create it with SQL. Then, you should be able to use the GUI to update that constraint.

- Add the foreign key constraints before you populate your tables to make sure your design is implementable. While populating your tables, you can temporarily disable foreign key checks (check this).
- You can use a tool like this (which will likely require some manual changes), write a script, or do it manually to populate your tables. You can simply use dummy data, meaning your data does not need to make sense, but using some realistic data can help you evaluate your design. Your database design must be able to work with real data.