INST733 – Database Design

Final Project Submission – See Canvas for the deadline

Report Requirements

As we approach the point in the semester when you will finalize and submit your projects, this document should help you conclude and present your work in a way that will best demonstrate how much you have learned about relational database design.

Your project needs to incorporate four aspects of database design work, and your work will be assessed based on these four aspects:
- logical design, (as demonstrated by the ERD you designed for your database),
- physical design, (as demonstrated by the physical database you created based on the ERD),
- sample data, and
- sample queries.

In addition to those four aspects, a report of your reflections on your project, including your critical evaluation of the database itself and the process through which you built it, will also be part of the assessment. Below are guidelines on how you should approach these aspects and include them in your project report and submission.

In addition to the guidelines in this document, please be sure to read the submission instructions which detail (among other things) the filename convention you must follow, as well as the provided grading rubric.

Logical Design: You will demonstrate your level of understanding of logical design principles and methods by submitting an ERD that is an accurate and up-to-date representation of the final structure of your database. Your database should include at least 10 tables. There can be exceptions to this rule, as you will see later in the document. If your design warrants fewer than ten tables, discuss the situation with the instructor. If the instructor supports your decision, you can compensate for fewer than 10 tables by including more than the minimum required number of views. (Refer to the CRUD section of the document for details). Once you are done with your work on your database, reverse engineer your database into an ERD, and submit it as part of your submission. Alternatively, you can update your ERD manually. The vital requirement is that the

ERD in your .mwb file matches the structure of your physical database. You must **also** include an image of your ERD in the written report, as a figure somewhere within the discussion of the logical design.

Physical Design: Your actual project database that resides on the MySQL server will be the showcase of your understanding of physical design concepts and methods. Once you are done with everything that you need to do on your database, including adding the sample data and the views you want to include, you should take a backup of your database in MySQL Workbench. Take the backup as a single, self-contained file, (**not** as a folder with multiple files, one for each table.) Make sure to include the `CREATE DATABASE` query. Once you take the backup, restore it back on some other computer to make sure that it actually works. (For example, team member A takes a backup; sends the backup file over to team member B, who then restores it on their computer and checks the database to make sure that everything is included in the backup, and that the tables, views and the data all look OK).

Sample Data: The sample data in your database will demonstrate your understanding of how information about real-world entities can be kept as data in your database.

In order to be able to effectively demonstrate that your tables, relationships and queries work, you will need to insert at least 20 sample records into each of your non-join (source) tables. If you add more than 20 sample records per table, that is fine; in fact, the more data, the better. The 20-row minimum does not apply to tables which by their nature have fewer than 20 records, such as a Payment_Types table, which probably would have fewer than 20 records by definition. Join (linking/junction) tables should have at least 50 records each; again, the more data, the better.

The sample data in your database should be as realistic as possible. It is not required that the data has perfect accuracy in terms of representing real-world information; for example, if you built a movie database and your sample data includes some incorrect information, such as a wrong production year for a movie, a wrong date of birth for a director, or minor incorrect matching between movies and actors/actresses, those are not deal breakers. However, your data should have "internal consistency"; for example, a name should look like a name, a date should look like a date, a phone number should look like a phone number, an email address should look like an email address, and so on. If you have "123" as a phone number, or "awerzew" as an email address, those will be problems that we will not overlook. Refrain from entering gibberish as data; for example, do not have the name of a person such as “SSSSvvv”. Feel free to ask us when you are in doubt about how far you should go to refine your sample data.

Also, make sure that your database backup includes all the sample data. If we restore your database and there is no data (or an insufficient amount of data) in any or all of your tables, we do not have any option other than assuming that there actually was no data (or an insufficient amount of data) in those tables.

CRUD: Sample queries will demonstrate your understanding of SQL operations. As far as CRUD operations go, we are interested in seeing your sample “read” queries; that is, queries that make use of the SELECT clause. We ask you to save your SELECT queries as views in your database, and include those views in the database backup (dump) you submit as part of your project. Here is what you need to include in terms of views/SELECT queries:

- Include at least five SELECT queries as views in your database, if you have ten or more tables in your database design, and you are a team with fewer than three members.
    - If you have three or more members, add two extra views for each member beyond the first two members. For example, if you have a three-member team, add two more views.
    - If you have fewer than 10 tables, and you are in a team with two or more members, add one more view to compensate for each table below 10. *If you are a “team” of only one person, you do not need to include any additional views to compensate for having fewer than 10 tables in your database.*
    - Feel free to consult with the instructors to clarify the number of views required to be included in your database, or see the table below.

Number of tables ➡	10	9	8	7	6
Team size ⬇					
One	5 views	5 views	5 views	5 views	5 views
Two	5 views	6 views	7 views	8 views	9 views
Three	7 views	8 views	9 views	10 views	11 views
Four	9 views	10 views	11 views	12 views	13 views

- These queries (views) should **not** make use of the wildcard character (*) unless it is unavoidable given the nature of the query. If that is the case, and you use the wildcard character (*), you need to include in your report an explanation about the need for such use in that query.

- At least four of your queries (saved as views) should involve multiple (two or more) tables, and thus involve JOIN clauses. (Requirement A)

- At least three of your queries should involve some form of filtering (WHERE, HAVING, etc.) (Requirement B)

- At least two of your queries should involve some form of aggregation over records (SUM, COUNT, AVERAGE, GROUP BY, etc.) These cannot be queries that simply count the number of rows in a given table, such as *SELECT COUNT(invoice_id) FROM invoices*. (Requirement C)

- At least one of your queries should involve a join (linking) table and both of its source tables. (Requirement D)

- At least one of your queries should use a subquery. (Requirement E)

- No two queries should be simple variations of each other. For example, avoid having two queries that display the same result set, but ordered in two different ways; or avoid having two queries that count the same set of rows using two different columns.

Obviously, one query may satisfy multiple requirements, and you may satisfy all of the requirements with only five queries. Copy and paste your queries into a text file, and submit that file with the rest of your project. (Use .txt as the file extension.) This does not relax the requirement about the views; you still need to include at least five of your queries as views in your database. However, by using the text file (.txt) you can submit more than five queries without necessarily adding them as views. Also, include in your report a table showing which queries satisfy which requirements. The table should look like this:

View name	Req. A	Req. B	Req. C	Req. D	Req. E
Query_xyz	X	X			
Query_pqr	X	X	X	X	
….	…				

Without this table, we will not be able to review and grade your sample queries/views, and you will lose points as a consequence.

Reflections (Project Diary and Report):

You are required to submit a written report as part of your project. The length of the report should be between 4-7 pages, including figures, images, tables, appendices, etc.. (1.5 line-spaced 12pt Times or equivalent font with 1" margins on all sides.)

Start with an introduction of your problem domain and motivation, and introduce your database explaining how it can fulfill the information need at hand. You may restate the information presented in your project proposal; however, make sure that your final report reflects the changes that you made to your design after the proposal was submitted. If there are notable changes to your design, major ideas you could not implement, or new ideas that emerged and were implemented as you worked on your database, those would make great discussion points for your report.

Another good discussion point is your "lessons learned." You may want to discuss about times when you got stuck working on the project and how you resolved the problem(s) and moved on. If you had multiple solution options and ended up choosing one, write briefly about all of those options and explain why you chose that solution over others. It is fine to discuss about problems you resolved through working with me, or peer coaches. You can also discuss about technical problems, such as those you experienced as a result of using MySQL Workbench, etc. In any case, try to keep the conversation on what you have learned.

Make sure that you cover all five major stages of your project work in your report: Logical design, Physical design, Sample data, CRUD operations (queries/views), and Feasibility analysis of implementing the database on an alternative platform/paradigm.

Also write about what you might add or change in your database if you were to work more on it. What would be some future improvements and extensions to your database? An integral part of this section is a feasibility analysis of implementing the database on an alternative platform/paradigm. In this portion of the discussion, you should provide reasons why a single server MySQL/MariaDB setup might be a limiting factor, and how you would overcome that limitation. For example, would you continue using the current paradigm and tool, but implement some expansion strategy, such as partitioning and replication, or would you use a different database paradigm, such as a NoSQL solution, or a combination approach?

Your final project report should at least include the following sections and subsections:

- Introduction
- Database Design and Implementation
  - Logical Design (*include a PNG of your ERD in this section*)
  - Physical Database
  - Sample Data
  - Views / Queries
- Changes from Original Design
- Issues Experienced During Development
  - Issues encountered
  - Solutions considered
  - Solutions chosen (*include reasons for those choices*)
- Lessons Learned
- Potential Future Work
  - Extensions within the current technology and design (*MySQL/MariaDB*)
  - Feasibility of alternative implementations (other relational considerations and/or non-relational solution, such as NoSQL approaches)