CSE 4/560 Data Models and Query Language
Semester Project

Instructor: Dr. Shamsad Parvin
TAs: Pengyu Yan, Yifan Yang, Shayantan Pal, Shahram Babaie,Jingchen sun

February 21, 2023

1 Overall Project Objective

Build up a database to demonstrate interesting searches. This project should be a team effort, and each team should contain three people. The deadline for team finalization is **Feb 16th 2023**. For each milestone, only the team leader (Member 1) needs to submit your work.

1.1 Tasks

The task details are as follows:

1. Select one interesting usecase domain, building your database using SQL. It should be relatively substantial, but not too enormous. Several project ideas are described at the end of this document. However, these ideas aim to support you start thinking, and you are encouraged to come up with your own choice of usecase. Please keep the following points in mind: a) while real datasets are highly recommended, you may also use program-generated "fake" datasets if real ones are too difficult to obtain; b) How will you use the data? What kind of queries do you want to ask? How is the data updated? Your application should support both queries and updates.

2. Design the database schema. Start with an E/R diagram and convert it to a relational schema. Identify any constraints that may be applicable in your usecase problem and implement them using database constraints. If you plan to work with real datasets, it is important to go over some real data samples to validate your design. Do not forget to apply database design theory and check for redundancies.

3. Create a sample database using a small subset by hand to facilitate debugging and testing because large datasets make debugging difficult. It is good for different scripts to automatically create/load/alter/update/destroy the sample database.

4. Acquire the large "production" dataset, either by downloading it from a real data source or generating it using a program. Make sure the dataset fits your schema. You might need to write programs/scripts to transform them into a suitable form for loading into a database for real datasets. For program-generated datasets, make sure they contain interesting enough "links" across rows of different tables to show the results of different Ad-vanced SQL queries learned in class.

5. You are required to make sure all of your relations are in Boyce-Codd Normal Form. Provide a list of dependencies for each relation. Decompose them if the tables are not in BCNF. If you decide to keep it in 3NF instead of BCNF, justify the decision for a particular relation. Your report for this milestone should contain a separate section with the details of the transformation from the initial schema to the final schema where the relations are in BCNF. **Note**: This is quite possible that your initial schema is already in BCNF, and in that case, you need to provide us the functional dependencies and convince us that the relations are already in BCNF.

6. Do you specifically run into any problem while handling the larger dataset? Did you try to adopt some indexing concept to resolve? Briefly describe the questions you faced and how you solved them.

7. Test your database with more than 8 SQL queries. You are supposed to design 1 or 2 queries for each inserting, deleting, and updating operation in your dataset. And please write select queries no less than 4 queries. Your select queries should be in different types of statements, for example, you can use "join", "order by", "group by", subquery, etc. Get your execution results and take screenshots to show them.

8. Query execution analysis: identify three problematic queries (show their cost), where the performance can be improved. Provide a detailed exe-cution plan (you may use EXPLAIN in PostgreSQL) on how you plan to improve these queries. You can find the EXPLAIN tool as the Figure 1.

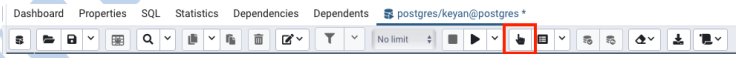

Figure 1: EXPLAIN tool

9. **Bonus Task [10]** Build a running website that links with your database to display/visualize query and query results. You can use any language

to implement your work. However, all the structures should be submitted in your second milestone, and the website address should be included in your final report.

1.2 Expected Result & Grade Distribution

1. Presentation and Demo: Record an 8-15 minutes video about your project's demo to UBLearn. Besides, you need to prepare a live presentation as well. It can be similar as your recording but our TAs will attend your live demo using zoom and ask questions related to your project. We will fix each checkpoint's presentation dates after the corresponding deadline.

2. Project Report(group): write the report and state clearly the contribution from each team member. It should be 5-8 pages in IEEE/ACM format(https://www.overleaf.com/latex/templates/ieee-conference-template-example/nsncsyjfmpxy) (`https://www.overleaf.com/latex/templates/acm-conference-proceedings-master-template/pnrfvrrdbfwt`). More details to follow within the milestone details.

3. Please submit your demo video, database SQL dump, and all other files within a zip file. The report should be **separately** uploaded as a pdf file.

4. Grade for Implementation/demo 60% and report 40%.

2 Deliverables

2.1 Milestone 1: Due date: 03/13/2023[100]

You are required to hand in a report that contains the overview of your project proposal. The overview can change slightly as we go over the course, but the central theme should be intact. The proposal should consist of two or more pages describing the problem you plan to solve, outlining how you plan to solve it, and describing what you will "deliver" for the final project. Your report should contain the following sections:

1. Project details: Name of your project, your team, and all team members, everyone's UB id(not the UB number);

2. Problem Statement: Describe the problem that your proposed database system will solve. Why do you need a database instead of an excel file?

3. Target user: Who will use your database? Who will administer the database? You are encouraged to give a real-life scenario;

4. E/R diagram: Draw an E/R diagram for your database and briefly de-scribe the relationships between different tables. (Do not draw the figure by hand, you may use any tools to design or generate your E/R diagram)

5. Task 1-4 should be complete, and you should start planning for tasks 5-7. The detailed description and demonstration of your work on each of these tasks should be presented in the Project description and demo presentation video.

6. Any code for downloading/scraping/transforming real data you have written for data acquiring should also be reported if applicable.

Save your report as *Milestone1.pdf* and upload it to UBLearn. Only the team leader (Member 1) needs to submit the file. For example usecases, please see section 3; while you may want to choose your own use-case, it should be equivalently detailed.

NOTE: Define a list of relations and their attributes.

1. Indicate the primary key and foreign keys (if any) for each relation. Justify your choice;

2. Write the detailed description of each attribute (for each table), its purpose, and datatype;

3. Indicate each attribute's default value (if any) or if the attribute can be set to 'null';

4. Explain the actions taken on any foreign key when the primary key (that the foreign key refer to) is deleted (e.g., no action, delete cascade, set null, set default).

2.2 Milestone 2: Due date: 05/01/2023 [100+10]

This is your final submission of the project. The complete report should be there.
Your complete report should contain:

1. Details of all the tasks required to perform in this project. Please highlight any new assumptions, E/R diagram, and list of tables (if they have changed since Milestone 1 that you have added/edited).

2. Create a file *create.sql* which will create all the tables in your database. Load these relations from data files (tab or comma-separated files). The tab or comma-separated files can be created by you (dummy values) or other sources. Create a *load.sql* file for bulk loading. Create a *readme.txt* file that states your data source. Put *create.sql*, *load.sql*, all the '.dat' files (or *.csv* files, or data files in any other format), and a *readme.txt* file into a sub-directory. If you generate and import your data through some scripts, you do not have to create *create.sql* or *load.sql*, but please also include a *readme.txt* file to describe how you built tables and imported data.

3. Demo Video, as mentioned in **Expected Result**

Save your final report as *Milestone2.pdf* and upload it to UBLearn. Only the team leader (Member 1) needs to submit the file.

Final Project Demo. You will need to present your system's working demo at the end of the semester. Instructions on how to sign up for the demo will be given during the second to last week of the class. You are also encouraged to stay in touch with the TAs (we will assign a TA for each team) to discuss your project and get their feedback on how to improve.

3 Example Application Scenarios:

Please follow the links for the details.

- IMDb makes their movie database available `https://www.imdb.com/interfaces/`.
- Data.gov `http://www.data.gov/` has a huge compilation of data sets produced by the US government.
- The Supreme Court Database (`http://scdb.wustl.edu/data.php` tracks all cases decided by the US Supreme Court.
- US government spending data (`https://files.usaspending.gov/database_download/`) has information about government contracts and awards.
- Federal Election Commission (`http://www.fec.gov/disclosure.shtml`) has campaign finance data to download; their "disclosure portal" (`http://www.fec.gov/pindex.shtml`) also provide nice interfaces for exploring the data.
- historical stock quote can be downloaded and scraped from many sites such as Yahoo! and Google Finance.
- You are allowed to use any open-source datasets.