WMDD 4921 Term Project

The purpose of this term project is to give you some practice designing, implementing, and using a moderately-complex relational **SQLite** database for a real-world application, with real data.

Please note that this is an ***individual*** project, not a team project, and the work you submit must be entirely your own.

Section 1: Reusing a Project from WMDD 4980

If you're currently taking **WMDD 4980 Project 2,** you may have the option of reusing your work in that course for this term project. If you're *not* interested in doing this, you can skip to **Section 2: Creating a New Project.** If you *are* interested, please check the following criteria and make sure ALL are met:

- You must meet with me ***in person*** to discuss a plan to reuse the project.
- Your Project 2 work must be from this term (no reusing old projects, please).
- The work must have been done almost entirely by yourself (at least 90%), on your own without help from your teammates.
- You must provide documentation of the work, including your role and responsibilities on the team, progress reports, and design documents.
    - For privacy reasons, if you have to share something your teammates worked on, please ask them first and have them email me directly to confirm their consent.
- The work must use **SQLite** and meet all the requirements of this project spec, particularly in **Section 4**. If there are some missing requirements, you may have to add a few things to fill the gaps. If your work from Project 2 differs too much from this spec, it may be better to create a new project instead.

If you're able to reuse your work from WMDD 4980 and choose to do so, skip to **Section 3: Getting Started.**

Section 2: Creating a New Project

If you are ***not*** currently taking Project 2 or you do not wish to use your work from that course, then your mission is the following: create an **SQLite** database for an application based on the city of Vancouver, using data from Vancouver's Open Data Portal! The application can be about whatever you want (as long as it's Vancouver related) but you should think about the following:

- Who are the expected users? (Examples: tourists, residents, city employees, businesses, etc.)
- What problem is the application meant to solve? (Examples: helping runners find water fountains, helping residents track the history of their neighborhood, etc.)
- What are some operations on the data that will need to be implemented? (Examples: changing the zoning classification of a neighborhood, recording a new public art piece, getting all addresses within a certain area, etc.)
- What data is necessary for your application? What data must you store?

If you're also taking **WMDD 4936 Full-Stack Web Development,** I ***strongly*** recommend that you also use the same application idea for that course's term project! You can get credit for the same application in both courses!

I recommend browsing the Vancouver Open Data Portal to get ideas for your application. For each data set, you can click Export and download the JSON data to see what's provided. (You may have to format it in a code editor to make it readable.)

Keep in mind that you only have a few weeks, so limit your scope accordingly. The amount of work you do should be equivalent to about **three weekly assignments.**

The dataset you choose will inform your process for designing a data model and schema. This data is provided in a (mostly) unnormalized format, and you'll have to normalize it. You'll also have to ingest the data with a Node.js script. You don't have to use all the data provided.

Some datasets may not be appropriate for this project, particularly if they're too simple or don't contain enough data. When you think you've found a dataset you like, I recommend skipping to **Section 4: Requirements** to make sure it will be suitable. You will have to create some of our own data or handle data created through use of the app, so that may fill the gaps sufficiently to meet the requirements.

Each dataset may be used by ***at most two students.*** If you choose the same dataset as another student, your application idea must be ***significantly different than theirs.*** To confirm this, read on to Section 3...

Section 3: Getting Started

Once you have your application idea, make a post in the discussion forum **Term Project Proposals** with the following:

- The name of your chosen dataset in the title of the post OR the name of your WMDD 4980 project.
- In the body of the post:
    - If you're using Vancouver Open Data:
        - a hyperlink to the **Information** tab of the dataset (for example: https://opendata.vancouver.ca/explore/dataset/property-tax-report/information/)
    - If you're reusing your WMDD 4980 project:
        - a link to the dataset you're using or explanation of the data you intend to collect.
    - A description of your application idea, including the intended users and functionalities.

Datasets and project ideas are claimed by students on a first come, first serve basis, so if you have an idea you like, please post it ASAP. I'll look at all the ideas and provide feedback if necessary, or simply reply **"APPROVED"**. I may ask you to choose a different idea if I think yours may not work. Once your project idea has been approved, read on to the next section...

Section 4: Requirements

Your project must have ***ALL*** of the following:

- A data model:
    - For students NOT reusing their 4980 project: you must add some new data features that aren't included in the dataset from Vancouver Open Data Portal, such as new attributes or entities. Probably this data will be needed to implement the logic of your specific chosen app.
    - At least four entities/tables with attributes, from your chosen dataset;
    - Normalized to third normal form;
    - Relations with keys and functional dependencies;
        - Provide a **PDF** document called **model.pdf** explaining your process for developing the above four things. Also explain how your application works (or how would it work if you were to build it.) Overall, this will serve as the documentation for your project.
    - An ERD diagram with relationships, ordinalities and cardinalities, and attributes;
        - Provide either a screenshot image or photograph of a drawing called **erd** (in JPEG or PNG format).

- A schema for SQLite:
    - At least four tables with primary keys and foreign keys;
        - Each column should have a type and sensible constraints;
    - At least one view;
    - At least one trigger;
    - At least one index to optimize search performance;
        - Provide a file called **schema.sql** for setting up the above schema. Comment your code thoroughly to explain each part of the schema and design decisions about constraints, etc.
- Queries:
    - This part will depend on a few things:
        - If this project doubles as your WMDD 4936 project, your queries will be implemented in your API. See that project spec for details.
        - Otherwise, you can simply make a file called **queries.sql** to write a selection of queries that demonstrate the functionalities of your app. Use comments to document your queries thoroughly, including where/how they would be used in the application.
    - Regardless of which above case applies to you, you must demonstrate the following:
        - Some kind of **filtering** with **WHERE**, **HAVING**, etc.
        - **Order** and **limit;**
        - **Aggregation** and **grouping;**
        - At least one **insert;**
        - At least one **join;**
        - At least one **update** and/or **delete;**
        - At least one **transaction.**

Section 5: Importing Data

If you're reusing your Project 2 data, this section may not apply to you; please provide all assets related to your process for ingesting data.

If you're creating a new application with data from Vancouver Open Data Portal, you'll have to import the relevant JSON files with a Node.js script, converting the JSON data and inserting rows into your SQLite database. I'll provide a template script and instructions to make this easier for you but you'll have to adapt it to your needs. The database file should be called **data.db** and the script to build it should be called **data.js.**

You don't have to use all data from the dataset. Vancouver's data is quite messy and inconsistent, but you don’t have worry too much about that; your own data model and schema should at least meet the consistency and design requirements from section 4.

Section 6: Attributions

As mentioned earlier, this is an **individual project** and all work must be done by you. That said, it's okay to consult resources. In your **model.pdf** file, please include a list of references to any sources you used, such as tutorials, articles, YouTube videos, or anything else. You must include enough detail that I can tell EXACTLY which work is yours and which came from the source. You must also provide a link to the exact source (For example, simply writing "YouTube" does not count as citing your source.)

Similarly, if you use AI tools to generate code, models and schemas, or any other assets, you must provide thorough documentation of your process. Please format this as a PDF file called **ai.pdf** with a point-form explanation of your entire AI workflow, including the problems you attempted to solve with AI, the programs you used (CoPilot, Cline, Cursor, etc.) the model (Claude, GPT-4.1, etc.) all prompts, rule files, MCP resources, and anything else that went into the process. History export files and other supporting files are helpful, but not sufficient on their own without your own explanation.

Section 7: Other Requirements

Here are some general things to keep in mind as you work:

- Document your process as much as you can, including with comments in your script files. If there's anything you think may be unexpected or confusing, please explain it.
- Format your code to make it easier to read.
- Use sensible and consistent names for columns, tables, etc.

Section 8: Milestones

Here are the milestones:

1. **Proposal**
    - **Date: ASAP**
    - Submit your proposal for approval to **Term Project Proposals.**
2. **Oral Exam and Code Review**
    - **Date: no later than my office hours on November 28th.**
    - When you've completed at least 75% of your work, meet me for a short in-person oral exam and code review. I will ask you **two questions** about your work and you must answer them without looking at it. This can double as one of your conferences, and you can do your oral exam for the WMDD 4936 project at the same time. (In that case, it will be **four questions** instead of two.) These meetings are first come, first serve so I recommend meeting me before the last week of classes to avoid the line. You are not guaranteed a meeting if time runs out.
3. **Final Hand In**
    - **Date: end of day on December 1**

Section 9: Hand In

Please gather the following files for hand in:

- **model.docx**
  - An explanation of your modeling and normalization process;
  - An explanation of your application;
  - Explanation of which new parts of the model you added that weren't in the original data set;
  - Attributions.
- **erd.png/jpg**
  - The Entity-Relationship diagram.
- **schema.sql**
  - The SQL script that builds the tables, views, etc.
- **data.db**
  - The actual database will all data.
- **data.js**
  - The Node.js script that converts data from the JSON format and stores it in the database. If you're using your 4980 project, include any other relevant files.
- **queries.sql** with a set of queries that meet the requirements in section 4 and demonstrate the features of your application.
  - If you're also using this project for 4936, you don't need to make a **queries.sql** file. Instead, include the Models folder from your API.
- The original JSON file from Vancouver Open Data Portal or whichever data source you used.
- **ai.pdf** (if you used AI-based tools)
  - Any other files necessary to support the explanation of your AI workflow.

Place all files in a folder called **project-firstname-lastname,** make sure no unnecessary files are included (like these instructions), zip the folder, and hand it in to Brightspace.

Section 10: Grading

- Oral exam and code review
  - [6 marks] three marks per question
- Model
  - [1 mark] entities and attributes from dataset, with new ones for your application
  - [1 mark] normalized
  - [1 mark] ERD
- Schema
  - [1 mark] tables with keys, constraints, and index
  - [1 mark] view
  - [1 mark] trigger
- Queries
  - [3 marks] roughly ½ mark for each type of query from the requirements
- Database
  - [1 mark] data successfully imported into database

Total: 16

Up to 50% may be deducted for the following:

- missing documentation, insufficient comments
- missing scripts or data files
- poorly-formatted code, disorganized or poorly-named files

Please ask me if in doubt!