CS 121 23wi Final Project Proposal

Due: Saturday Feb. 18th, 11:30PM

(NOTE: we got an extension from El until the end of Tuesday 2/21)

For the final project, you may choose to work with up to one other person. For partner-finding, use the #partner-search Discord channel to share your interests, any ideas, and preferences for working with a partner. Tuesday's lecture (02/14) will be optional worktime to find partners and ask about any questions with El around; we encourage you to attend to aim for finalizing a project idea, a partner, and any dataset/scoping questions you have. You are welcome to remove any suggestions/notes from this proposal in your proposal, as long as you're required components are clearly in the order specified (we've highlighted text for you to answer in blue for ease).

Note: If working with another student, only one of you needs to submit the required files to CodePost, but ***both of you should have equal contributions; i.e. you should ideally work through this document together.*** *For ease of grading please have the other student submit a simple* **collaboration.txt** *file to their CodePost submission in the following format (failure to do so may result in deductions):*

```
Names: <student1>, <student2>

CodePost emails: <email1>, <email2>
```

For full credit, we are looking for a robust proposal demonstrating careful consideration of the motivation, design process, and implementation plan for your application; you should expect to spend 3-4 hours minimum on your proposal (finding a dataset may take the longest). For each part of this proposal that requires your response, you should have 2-3 sentences (bullet points are encouraged as well) justifying your answers. We strongly encourage you to include as much detail as you can in this proposal (this is your only assignment this week) and you can include any diagrams and SQL brainstorming (e.g. a start to your DDL) with your submission on CodePost. If you want to add any additional planning, you can include a diagram as a PDF/PNG/etc. and/or .sql files, though not required. You can also include a starter Python program with function stubs for a command-line implementation you will complete for the Final Project (without SQL integration).

Summary of Files to Submit (max 3MB, reach out to El if you have trouble with this limit):

- **proposal.pdf (a copy of this document filled out)**
- ***(optional) diagrams/sketches/brainstorming not otherwise required; you can include these at the bottom of your proposal document.***
- ***(optional) app.*.py funtion stubs***

The advantage of adding these in your proposal is to get you started in advance, and also to get feedback from the staff on your design and implementation so far.

Student name(s): Kyla Yu-Swanson, Riya Shrivastava

Student email(s): kyuswans@caltech.edu, rshrivas@caltech.edu

Database/Application Overview

In this proposal, you will be "pitching" your project, in which you have some freedom in choosing the domain of with a structured set of requirements that bring together the course material in a single project, from design to implementation to tuning. Keep in mind that unlike CS 121 assignments, you are free to publish/share your project, which can be useful for internship or job applications. In terms of scope, you should shoot for 8-12 hours on this project, though you are free to go above and beyond if you choose.

First, what type of database application are you planning on designing? Remember that the focus of this project is the database DDL and DML, but there will be **a small Python command-line application** component to motivate its use and give you practice applying everything this term in an interactive program; you can find a template from last year here, which may be adjusted slightly before the Final Project is released, but gives you an idea of the breakdown. Don't worry too much about implementation at this step, and you can jot down a few ideas if you have more than one. Just think about something you would like to build "if you had the data" which could be simulated with a command-line program in Python. Your command-line program will start with a main menu with usage options for different features in your application. Staff are here to help you with scoping!

Here's a list of some application ideas to get you started. These encompass most of the applications within the scope of this project we anticipate students might be interested in, but if there's a different application that meets the requirements for your DB schema and implementation, you are welcome to ask!

Proposed Database and Application Program Answer (3-4 sentences) :

We plan to make a recipe and ingredient database. This will be useful to people trying to decide what to cook with the ingredients they have or to find ingredients that can be used for the most recipes. The admin section of the database will also allow chefs to add new recipes and ingredients.

Next, where will you be getting your data? We will post a list of datasets to get you started, but you can find many here and here (look for ones in CSV file(s), which you will break into schemas similar to the Spotify assignment; we can also help students convert JSON to CSV if needed). You are also welcome to auto-generate your own datasets (e.g. with a Python script) and staff are more than happy to help you with the dataset-finding process, which can take longer than the rest of the starting design process.

Data set (general or specific) Answer:

We will be using data from this repository: https://github.com/cweber/cookbook. It needs some cleaning since many of the quantities are messed up and displayed as dates instead of fractions. We will also be making a table to categorize the categories (eg. cake is a subcategory of dessert) so that users can search for a type of food in more and less specific ways. This will require some modification of the csv files. We've included the csv file we started modifying as recipes-project.csv. This file will later be used to create all the table specific csv files for importing.

Who is the intended user base and what is their role? You will need to have at least one client usertype, and one admin. These different users may have different permissions granted for tables or procedural SQL. Consider clients as having the ability to search ("query") data from the command line, submit requests to insert/update data, etc. In 22wi, some students had a separate client .py file and admin .py file, with admins having admin-related features, such as approving requests, inserting/updating/deleting information, querying, etc.

***Client user(s) Answer*:**

The intended user base will be people learning to cook. These people can query the database to find information, but cannot change the tables.

Admin user(s) Answer:

The intended admin users will be chefs that create new recipes and want to add their recipe to the database.

Required Features

The full specification of the project will outline the requirements of the project, but you remember you should aim to spend 8-12 hours (it replaces the final exam and A8), depending on how far you want to go with it. The time spent on finding or creating a dataset will vary the most. For the proposal, you will need to brainstorm the following (you can change your decisions later if needed).

E-R Model: There will be an ER model component, but it is not required for the proposal.

DDL: What are possible tables you would have for your database? Remember to think about your application (e.g. command-line functions for user prompts to select, add, update, and delete data, either as a client or an admin user). To make this step easier, thinking about the menu options your application provides, as well as the queries you might want to support, is helpful to narrow down the information you'll want to model. At minimum, you must provide a general listing of attributes and possible types; you can provide these on a high-level (e.g. bullet points) or actual DDL with CREATE TABLE statements in an optional **`setup_proposal.sql`** file attached in your submission. The more you provide, the more we can help provide feedback. Include at least 4 tables in your response, as well as any design decisions you may run into in your schema breakdown.

Answer: see setup_proposal.sql. We struggled with how to represent the varied number of ingredients for each recipe, so there is an implementation with JSON and an implementation that doesn't have a primary key. We will likely use the latter because it is easier to work with.

Queries: Part of the application will involve querying data in a way that would be useful for a client (e.g. searching for "Puzzle" games made in the last 5 years, ordered by year and price in a Video Game Store database, or finding all applicants who are pending for an adoption agency). Identify at least 3 queries that would make sense in a simple command-line application. In your answers, provide a brief description of the query or pseudocode, as well as the purpose in your application. These will likely be implemented as SQL queries within Python functions, wrapping your SQL queries (the methods of which will be taught in class). You are welcome (and encouraged) to add more, though not required.

Answers:

1. Find all ingredients in pumpkin pie. This is an example of finding the ingredient_list variable/subquery that is used later.

    a. Pseudocode:

    ```
    SELECT ingredient_id
    FROM recipe JOIN instructions USING(recipe_id)
    WHERE recipe_name = "pumpkin pie";
    ```

2. Find all recipes that use the ingredient "pumpkin spice". This is an example of a query that would help users find recipes that they can make with the ingredients they already have. Note that ingredient_list will be obtained with a separate query.

    a. Pseudocode:

    ```
    SELECT recipe_name
    FROM recipe JOIN instructions USING(recipe_id)
    WHERE "pumpkin spice" IN ingredient_list;
    ```

3. Find all ingredients in recipes of the category "Cake." This is an example of when a user would try to find recipes when learning a certain cooking style.

    a. Pseudocode:

    ```
    SELECT ingredients
    FROM recipe JOIN instructions USING(recipe_id)
    WHERE category = "cake";
    ```

4. Find the recipes with less than five ingredients. This is an example of when a user wants to cook a simple dish. Note that ingredient_list will be obtained with a separate query.

a. Pseudocode:

```
SELECT recipe_name, num_ingredients
FROM (SELECT recipe_name, COUNT(ingredients_list) as num_ingredients
        FROM recipe JOIN instructions USING(recipe_id)) as subquery
WHERE num_ingredients < 5;
```

Procedural SQL: You will need to implement at least 1 each of a UDF, procedure, and trigger in your project (we haven't seen triggers yet, so you don't need to provide one in your proposal). Otherwise, identify at least one UDF and one procedure here. For each:

1. What is the name of the function/procedure?
2. Why is it motivated in your database? Consider the differences discussed in Lecture 10 for UDFs, procedures, queries, temporary tables, and views. In your Final Project, we'll be looking for you to demonstrate an understanding of appropriate design decisions here (and we're happy to help discuss any trade-offs)

Consider some examples in class/HW, such as logging DML queries, adding an extra layer of constraint-handling (e.g. the overdraft example from lecture), etc. For procedures, remember that these can be called in an application program written in a language like Python or Node.js, so these are especially useful to avoid writing queries in such application programs, *especially* SQL that performs **`INSERT/DELETE/UPDATE`** which you do not want to leave to an application user. Remember that you can also set permissions for different users to access tables and procedures. In your Python program, you can connect to your database as different users defined in your database (similar to the Node.js program El demo'd a while back).

Answers

UDF(s):

1. random_recipe

2. This user-defined function returns a random recipe that contains a given ingredient. This would be helpful to all users looking to find any random recipe with a specific ingredient, if, for example, they are trying to clean out their fridge! This function will have one specific parameter passed in (the ingredient), and will determine the corresponding ingredient_id and make use of the built in SQL RAND() function to return a recipe_id containing that ingredient. It would be useful for this to be a UDF, since we would be able to carry out row set operations and JOINs.

Procedure(s):

1. count_recipe_ingredients

2. This procedure returns a single value, which is the number of ingredients in a specific recipe. This is helpful to users as they plan out the scope of the recipe they are creating. For example, if a user is going on vacation and wants to cook some recipes with a limited number of ingredients, they can use this procedure to determine how many ingredients their chosen recipes have. This way, they can find the recipes of their liking with the least number of ingredients and then pack ingredients accordingly. Since this returns a single value, it is helpful to implement as a procedure.

~~**Trigger(s).**~~ ***(not required for 23wi proposal)***

~~**Database Performance:**~~ ***(not required for 23wi proposal, but left in for a preview of the corresponding Final Project component)*** *At this point, you should have a rough feel for the shape and use of your queries and schemas. In the final project, you will need to add at least one index and show that it makes a performance benefit for some query(s). You don't need to identify what index(es) you choose right now (that comes with tuning) but you will need to briefly describe how and when you would go about choosing an index(es). Refer to the material on indexes if needed.*

~~**Performance Tuning Brainstorming:**~~

"Stretch goals"

If you are particularly eager for a certain application (have your own start-up in mind?), it is easy to over-scope a final project, especially one that isn't a term-long project. You can list any stretch goals you might have "if you had the time" which staff can help give feedback on prioritizing.

Answer:

A stretch goal could be adding tables that include prices of ingredients and a rating system for recipes. For the prices, it would likely be good to write a table ingredient_id and a price_per_ounce attribute. This data would probably need to be generated somehow since we haven't been able to find a dataset for it. For the rating, we would also make a new table. These tables are both included in proposal-setup.sql with a note. Another potential idea is to add nutritional information or group categories for if a food is mainly a carb, protein, or fat since that would be simpler. This isn't added to the proposal-setup.sql file yet, since we would need to discuss it a bit more. Another idea is to add some kind of functionality to check for allergy concerns or dietary preferences. This also hasn't been added in the proposal-setup.sql file.

Potential roadblocks

List any problems (at least one) you think could come up in the design and implementation of your database/application.

Answer:
This was mentioned briefly above, but we considered a few options when determining how to create the ingredient_amounts table. Currently, the user would need to create a query to obtain all the ingredients belonging to a single recipe_id, since each ingredient/quantity is stored with a single recipe_id (and that recipe_id is repeated for however many ingredients that recipe has). This schema made the most sense to implement, but we want to ensure that it is not difficult for a user to clearly retrieve all the ingredients and corresponding quantities needed for a specific recipe, since right now it is not totally straightforward.

Additionally, since we have to process the original data and possibly generate new data for the reach goals, this could be a source of frustration since it is tedious, although some parts of it can be sped up with automation.

COLLABORATION

For projects with partners, this section is required (if not, you can leave it out, or note that you are decided yet). Both students should provide a brief summary of their planned workload distribution, method(s) of collaboration, and 1-2 points you are most interested in the project. You may also clarify any concerns/confidence for your partner work here. Feel free to provide a paragraph or bullet points; we're looking for you to have thought this out and discussed your plan for collaboration at this step.

We plan to split the work for making CSV files to import to the tables outlined by the DDL. This initial data processing/generation will likely take the most time. We are splitting up some of the files that need to be submitted and collaborating on the more intensive command line and reflection sections.

Kyla Yu-Swanson

- Setup-passwords.sql - password management functions + table + procedure
- Setup.sql - setup the tables' schema and at least 1 index
- Queries.sql - demonstrate SQL queries that can be used

Most interested in: User functionality. I want to make sure it's easy to access relevant information in the database, so I will be careful with writing the schema we've discussed and processing the data so that the values are clear.

Riya Shrivastava

- load-data.sql - statements to load database from our CSV files
- Grant-permissions.sql - create and grant user permissions
- setup-routines.sql - stored routines/triggers

Most interested in: Brainstorming/writing relevant routines and triggers that will be helpful to users and take advantage of database functionality, and working on the command line application to see how the whole application can be tied together.

Shared

- Split work for making CSV files to import to tables + data processing/generation
- app.py - work intensive, so we will each work on the parts that are related to what we worked on in other sections
- Written parts: reflection (including ER-diagrams, Functional Dependency Theory, Relational Algebra)
- README

Methods of Collaboration

- Discussion + working through any roadblocks/questions together
- Github repo
- Live share / pair programming if needed, live work sessions

---

Open Questions

Is there something you would like to learn how to implement in lecture? Any other questions or concerns? Is there anything the course staff can do to help accommodate these concerns?

Answer:

I think it would be useful to talk about using JSON data. I've used it before in PostgreSQL databases in an internship I did, so it would be useful to cover. Relatedly, it would be good to cover alternatives to using JSON lists for when there is a varied number of items of a value that need to be stored per row (for example, in a recipe, there are several ingredients).

Have fun!

Optional Brainstorming/Sketches/Other Notes

This is an optional section you can provide any other brainstorming notes here that may help in the design phase of your database project (you may find it helpful to refer to the early design phase in your Final Project Reflection).
