Choose Your Own Adventure (Final Project)

Stats 167 Lec 1

Spring 2025

Deadline

The project must be submitted to the portal on Bruin Learn by **Friday, June 13, 11:59pm** (the end of the quarter). **There is no grace period.**

General Guidelines

Any and all course material may not be posted online or shared with anyone at any time without explicit written permission by the instructor (Michael Tsiang), even after the quarter is over. Failure to comply is a breach of academic integrity.

For Option 1: You must treat this project like a take-home final exam or a technical interview. In particular, the project is open note, and you can use general SQL documentation or articles to help with understanding specific SQL functions, but **you are not allowed to collaborate with others or use any AI tools**.

For Options 2-4: You are allowed to collaborate with others to find resources or articles that you find useful/insightful, but each student must have independently completed projects. Any collaborations must also be stated in a citation/collaboration statement included in your project.

Project Specifications

This course has introduced fundamental concepts of data management, data engineering, database modeling, relational and non-relational databases, and SQL. The goals of the course are to build your knowledge and provide an opportunity to practice building skills to serve as practical preparation for applications, interviews, and careers in data science.

Since each student's motivation and interests differ, we want to provide you with the opportunity to practice and develop further the skills of your choice. You have **four different project options** to choose from, based on your interests and goals.

What to Submit

All options should include:

- SQL code (and R and/or Python code if used)
- A short write-up, report, or slide presentation (depending on the option) in PDF format
- Any supporting files (e.g., SQL scripts to create data, links to the data, diagrams)

Option 1: Practice SQL Interview Questions

The purpose of this project is not only to demonstrate your SQL knowledge but also practice communicating your thought process to others. This is intended to reflect what is expected in data-related job interviews, where you often are asked to explain your reasoning before or while writing code. Communication is also crucial when collaborating with teammates or writing production code on the job; your logic and code should be easy for others to understand and build on.

Your tasks

- Write SQL queries for a set of provided interview-style problems that test your understanding of SQL logic and query structure. The questions are attached at the end of this project document.
- Include short explanations for your approach and any assumptions you make.

Guidelines for your work

- Organize your code, string, and output in a clear way that is easy to follow.
- Write SQL code that is correct, concise, and well-formatted.
- Include brief but thoughtful explanations of your reasoning and results.

Example question and response

Here is an example of describing your thought process:

Example: Write a SQL query to find all employees in the `employees` table whose `salary` is greater than the average salary in their department. Assume that the table has `employee_id`, `department_id`, and `salary` columns.

```sql
SELECT employee_id, department_id, salary
FROM employees AS e
WHERE salary > (
    SELECT AVG(salary)
    FROM employees
    WHERE department_id = e.department_id);
```

Logic:

- The key steps are to calculate the average salary in each department and then compare each individual's salary to their department average.
- A straightforward way to compare is to filter for each individual, which we can do using the `WHERE` keyword.
- When filtering by the salary, we can write a subquery that calculates the average salary within each department.
- The line `WHERE department_id = e.department_id` retains employees in the same department as the current employee when calculating the average.
- Thus, the main query then selects employees whose salary exceeds the average salary of their respective department.

Option 2: Analyze a Real Database

In this project, you will be understanding a database by writing SQL queries to extract and manipulate data, followed by performing exploratory data analysis (EDA). Since the data is stored across multiple tables, you will need to write SQL queries to gather information and perform analyses like comparing group averages, calculating changes over time, and identifying outliers to find trends, patterns, and insights.

After querying and cleaning your data in SQL, you may use R and/or Python to conduct further analysis (e.g., statistical tests, visualizations, or modeling). This project simulates a common real-world data workflow: data wrangling in SQL, analysis and communication in R/Python.

Your tasks

- Choose a real-world database (e.g., one of the medium- to large-sized ones from the CTU Relational Database Repository or Kaggle) of your own interest to analyze.
- Propose at least 8-12 exploratory data analysis questions.
- Write SQL queries to answer your chosen questions. Your queries should demonstrate meaningful exploratory analysis, not just simple lookups.
- Your queries must include the use of:
  - Joins
  - Subqueries
  - Window functions
  - Common Table Expressions (CTEs)
- Summarize your findings in a short report. You may include R and/or Python code, visualizations, and interpretations.

Guidelines for your work

- Organize your code, output, string, and figures in a clear way that is easy to follow.
- Write SQL, R, and/or Python code that is correct, concise, and well-formatted.
- Choose questions that are relevant to the data and demonstrate your understanding of the database relationships.
- Include representative output for your queries, e.g., summary tables, small data previews, and plots. Not every query needs to have output shown, but key ones should.
- In your report, provide sound interpretations:
  - Discuss patterns in graphs or summary tables.
  - Comment on statistical significance (e.g., p-values).
  - Assess model performance (e.g., accuracy or goodness of fit) if applicable.
  - Your conclusions do not need to be perfect or robust, but they must be supported by the data and clearly communicated.

Example databases to choose from

- Airline data: https://relational.fel.cvut.cz/dataset/Airline
- F1 races (1950-2017): https://relational.fel.cvut.cz/dataset/ErgastF1
- IMDb: https://relational.fel.cvut.cz/dataset/IMDb
- NSF grants: https://relational.fel.cvut.cz/dataset/Grants
- Northwind; small business operations: https://github.com/jpwhite3/northwind-SQLite3
- Chinook; digital music shop: https://github.com/lerocha/chinook-database/blob/master/ChinookDatabase/DataSources/Chinook_Sqlite.sqlite
- Netflix content: https://github.com/lerocha/netflixdb
- Sakila; DVD rental: https://github.com/jOOQ/sakila
- NBA database (up till 2022/2023): https://www.kaggle.com/datasets/wyattowalsh/basketball
- Pitchfork music reviews: https://www.kaggle.com/datasets/nolanbconaway/24169-pitchfork-reviews

Other databases can be chosen, but you must email the teaching team (the TA and instructor) for approval to assess the suitability of your choice.

Option 3: Design and Query Your Own Database

For this project, you will design a small relational database on a topic of your choice, populate it with sample data, and write SQL queries to explore it. This is intended to reflect what is expected of data engineers and data analysts, who are tasked with structuring raw information into usable formats and generating insights.

Your tasks

- Describe a real or hypothetical scenario where using a relational database makes sense. This could be based on a hobby, business idea, organization, or any topic of interest. Explain briefly why a relational database is a good fit compared to storing data in a single table or spreadsheet.
- Design a database model based on your scenario.
  - Define at least 3 tables, including all fields/columns, and specify the relationships between them.
  - Resolve any many-to-many relationships.
  - Include an entity-relationship diagram (ERD) or visual schema.
  - Identify the primary keys and foreign keys for each table.
- Create and populate your tables using simulated or sample data.
  - The dataset does not have to be large, just realistic enough to test relationships and queries.
- Write SQL queries to explore the data.
  - You should write at least 6-8 meaningful queries that demonstrate understanding of your schema and answer relevant questions.
  - Your set of queries should include at least one of each of the following:
    - Joins
    - Subqueries or Common Table Expressions (CTEs)
    - Optional: Window functions, if appropriate
  - Summarize your findings and reflect on what your queries show about the data.

Guidelines for your work

- Organize your code, output, string, and diagram(s) in a clear way that is easy to follow.
- Write SQL code that is correct, concise, and well-formatted.
- Choose queries that are relevant to your scenario and demonstrate your understanding of the database relationships.
- Your queries should demonstrate that your schema is functional and appropriately normalized (i.e., minimal redundancy, appropriate use of keys).

Option 4: Explore an Advanced Database Topic

For this project, you will explore a new topic that was not fully covered in class and create a tutorial to teach it to others. This is intended to be an opportunity to deepen your knowledge on a topic of your interest, demonstrate your ability to learn independently, and practice explaining complex ideas clearly, all valuable skills for technical interviews and data-related jobs.

The tutorial should be structured so it can serve as a self-guided learning resource for another student. Your final project should be presented as either a knitted R Markdown report or presentation slides in PDF format.

Your tasks

Choose a topic that interests you. Your tutorial should include:

- Explanation of the topic
  - What the topic is and why it matters
  - Real-world use cases and applications
  - Key concepts and terminology
  - If relevant: Advantages, limitations, and comparisons with alternative methods
- Implementation with working examples
  - Use a database of your choice (e.g., one from class, a public repository, or a small one you create)
  - Demonstrate general syntax and typical arguments
  - Showcase specific features (e.g., `DATETIME` functions have different units for time) and functionalities (e.g., window functions allow you to offset `X` number of rows)
  - Include both correct use and common mistakes
- References and further learning
  - Include links to articles, documentation, or tutorials that helped you learn the topic
  - Provide resources for others who want to learn beyond your tutorial

Guidelines for your work

- Organize your tutorial with clear section headings and explanations before code (similar to the lecture slides).
- Write SQL examples that are correct, concise, and well-formatted.
- Demonstrate a solid grasp of the topic, not just definitions but also use cases, subtleties, and best practices.
- Assume the reader has taken this course: Your tutorial should be self-contained and understandable without external help or information.
- Cite all external materials or examples (blog posts, articles, documentation, etc.) with clear in-string links and references (not just a bibliography at the end).

Example topics to choose from

Choose a topic beyond the course content. Possible topics include:

- Advanced SQL topics
  - Data Control Language (DCL): User roles, permissions, and security
  - Transaction Control Language (TCL): Transactions, rollback, and isolation levels
  - Recursive CTEs
  - Stored procedures and triggers
  - Query optimization: Indexes, `EXPLAIN`, execution plans, and performance strategies
- Advanced NoSQL topics
  - Aggregation pipelines in MongoDB
  - Denormalization in Cassandra

Other topics can be chosen, but you must email the teaching team (the TA and instructor) for approval to assess the suitability of your choice.

Grading Criteria

All options will be graded using the same core rubric. Below are several guidelines to help understand our expectations for your project.

Technical Correctness

- Code is free of syntax errors and will run as expected
- Queries produce accurate results based on problem specifications
- Implementation demonstrates appropriate use of SQL concepts
- For relevant options: Data modeling decisions are valid and consistent with the scenario
- Terminology and technical explanations are correct

Depth or Complexity

- Project demonstrates thoughtful application of concepts beyond simple use cases
- Uses a range of relevant tools and techniques (e.g., joins, subqueries, window functions, and CTEs)
- For relevant options: Asks meaningful questions or addresses non-trivial scenarios
- For Option 4: Explains nuances or subtleties of the topic, not just definitions or syntax

Clarity and Communication

- Project is well-organized and easy to follow
- Code is clean, readable, and well-formatted
- Explanations are clear, logical, and connected to the code or analysis
- For Option 4: Project could be understood by another student in the class

Effort and Completeness

- All required components for the chosen option are clearly addressed
- Demonstrates effort to go beyond the minimum (e.g., well-written explanations, insightful reflections, thoughtful design, well-chosen database or topic)

Option 1: Practice SQL Interview Questions

Question 1

Suppose you are given the table below:

`sales`

column name	type
product_id	integer
sale_date	integer
amount	float

Compute the running total of sales for each product over time.

Question 2

Suppose you are given the tables below:

`products`

column name	type
id	integer
name	integer
category	string

`order_items`

column name	type
product_id	integer
quantity	integer
price	float

For each category, list the top 2 products by total revenue.

Question 3

Suppose you are given the tables below:

`orders`

column name	type
id	integer
customer_id	integer

`order_items`

column name	type
order_id	integer
product_id	integer
quantity	integer
price	float

Find customers who always ordered the same product.

Question 4

Suppose you are given the table below:

`marathon_times`

column name	type
runner_id	integer
race_number	integer
finish_time	float (in minutes)

Find all runners whose finish times strictly decreased in every consecutive race they have run. For each runner, assume `race_number = 1` is their first race, `race_number = 2` is their next race, etc. Include runners with only one recorded race in your result.

Question 5

The language learning app Duolingo wants to calculate the day-one retention rate of its users. A user is considered retained if they log in exactly one day after their first-ever login on the app.

Return the proportion of (day one) retained users out of the total number of users.

`user_logins`

column name	type
user_id	integer
login_date	date

Question 6

Ticketmaster wants to identify all pairs of available adjacent seats for an upcoming show at the Pawnee Amphitheatre.

Two seats are adjacent if one is listed as the left or right neighbor of the other in the seat map.

Return distinct pairs of available seats with the smaller seat number in the first column and the larger seat number in the second.

`availability`

column name	type
seat_number	integer
is_available	integer (0 or 1)

`seatmap`

column name	type
seat_number	integer
seat_left	integer
seat_right	integer

Question 7

Spotify wants to identify the genres of the most-streamed artists. An artist's total stream count is calculated as the sum of streams across all of their tracks. If multiple artists are tied for the top total stream count, include all of them.

Return the genre(s) of these top-streamed artist(s), sorted alphabetically.

Note: Assume each artist has only one associated genre.

`spotify_streams`

column name	type
artist_name	string
track_name	string
stream_count	integer

`artist_info`

column name	type
name	string
genre	string

Question 8

Find the number of items purchased by each Nordstrom member under different promotions. The promotions are "Full Price", "Sale", and "Clearance". Return the user ID and the counts for each item type.

`nordstrom_purchase_items`

column name	type
user_id	integer
product_id	integer
total_sales	float
purchase_date	timestamp

`nordstrom_items_labels`

column name	type
id	integer
promotion_type	string

Question 9

One metric to gauge the popularity of a restaurant is the number of reviews. Your task is to find the most popular restaurant on Yelp in 2025 so far. Note that an user can write multiple reviews, so only count the number of unique users. If a tie occurs, list all business IDs.

`yelp_reviews`

column name	type
user_id	integer
business_id	integer
review_id	integer
review_date	timestamp

Question 10

Suppose you are given the following tables on from Uber Eats:

`ue_orders`

column name	type
id	integer
customer_id	integer
seller_id	integer
order_timestamp	timestamp
amount	float
city_id	integer

`ue_cities`

column name	type
id	integer
name	string
timezone	float

`ue_partners`

column name	type
id	integer
name	string
category	float

Which partners have "fresh" in its name and are located in Los Angeles? And what is the average order amount from these businesses in the past 90 days (excluding today)? Output the names and order by average order amount in descending order.

Question 11

Now, let's take a closer look at the orders themselves:

`ue_delivery_orders`

column name	type
delivery_id	string
order_placed_time	timestamp
predicted_delivery_time	timestamp
actual_delivery_time	timestamp
delivery_rating	float
driver_id	string
business_id	string
customer_id	string

`ue_orders_value`

column name	type
delivery_id	string
sales_amount	float

Uber is reviewing its 2024 monthly sales on its food delivery platform. For each month, calculate which percentage of restaurants have reached at least $150 or more in monthly sales.

Note: If an order has no value for `actual_delivery_time`, it has been canceled and therefore does not count towards the sales.

Question 12

The table below contains customer data from Delta Airlines.

`delta_customers`

column name	type
id	integer
user_id	integer
flight_date	timestamp
flight_number	string
arrival_time	timestamp
actual_arrival_time	timestamp
total_sales	float

List the IDs of Delta customers who have taken at least 6 flights in 2023 and 2024. If there is no actual arrival time, this means the flight was cancelled and the customer did not take this flight.

Question 13

Amazon would like to categorize their users into Low/Medium/High tiers based on their "cart size". The average cart size is defined as `total sales amount/ number of transactions` for each user. The different tiers are defined as:

- If average cart size is more than $50, then Tier is "High".
- If average cart size is between $25 and $50, then Tier is "Medium".
- If average cart size is less than $25, then Tier is "Low".

The customer spending data are stored in the below tables.

`amzn_transactions`

column name	type
customer_id	integer
store_id	integer
transaction_date	timestamp
transaction_id	integer
product_id	integer
sales	float

`amzn_stores`

column name	type
store_id	integer
store_brand	string
location	string

Summarize the total number of transactions, total sales, the number of unique customers, and average cart size, grouped by store brand and tier for 2025 so far.

The final result should include the brand, tier, number of customers, total transactions, total sales, and average cart size.