CS 279: Capstone Project Data Science

Chat with the Map

Analyzing data on geographic maps has been a significant practice for over 170 years to study different kinds of human phenomena and support many applications in various domains. In the past two decades, interactive digital maps have revolutionized map-based data analysis by significantly improving the user experience and enabling end users to interactively browse and zoom different parts of the data, and link different data sources in interactive cooperative views that were not possible before with traditional maps. Today, in the era of Language Models (LMs) and Large Language Models (LLMs), we have a unique opportunity to have a second revolution in the user experience for digital map-based data analysis by enabling end users to interact with maps through their natural languages, replacing the existing way that requires the user to learn expert tools or complicated interfaces. Your project is one step towards this second revolution.

Task

You will build a lightweight "chat-with-the-map" web application that integrates a light LM with a backend spatial database. Users type natural-language spatial questions, the system converts them to PostGIS SQL, executes the query, and returns (a) a textual answer and (b) an interactive map visualization (if applicable). **Your application must log every intermediate step.** For example, a user asks **"**Which US counties have area < 100 mi^2?**" and immediately sees** both the **answer** and its **map visualization** with the exact SQL that produced it.

You are expected to use state-of-the-art Natural Language (NL) to SQL techniques, which typically provide a match of ~70% within a few seconds (3-9 seconds) of end-to-end latency for most of the use cases. In all cases, 100% of answers carry a citation to the executed SQL statements. You are not required to train large models from scratch, perform multi-dataset fusion, or build heavy custom front-end frameworks.

Project Data

Definition: US County: A US county is an administrative and political subdivision of a US state, functioning as an intermediate level of government between the state and cities.

Dataset

US Counties dataset extracted from TIGER files of US Census data. Download the dataset on https://drive.google.com/drive/folders/1AHEJB3e1OyIwOE0iy2047ol8BEt84Cu6?usp=sharing

Sample Test Questions

Below is **just a sample of test questions** and NL prompts for your web application. The actual test will include other questions, all in English natural language, on the same dataset.

1. Show me [Madison County, Idaho]

2. Visualize me [Madison County] in all states
3. Tell me, how many counties are called [Madison County]?
4. What are the [three] most frequent county names in USA?
5. Show me all [provinces] in [Canada]
   (out of the geographic scope, should be caught as an error)
6. List all counties in [Florida].
7. List all counties in [WA].
8. Which counties start with [‘San ’] in [California]?
9. Can you list all counties whose name equals their state?
10. Which counties have multi-word names in [Minnesota]?
11. Which counties in [California] touch [Nevada]?
12. What is the land area (km^2/mi^2) of [Riverside County, California]?
13. Rank all counties in [Arizona] by area.
14. Which county in [NC] has the smallest area?
15. Which counties nationwide have [area < 100 mi^2]?
16. Return the perimeter length of [Orange County, California].
17. Which counties in [South Dakota] have [perimeter > 800 mi]?
18. Which counties have holes (interior rings) in their geometry?
19. Which counties are multipart (non-contiguous MultiPolygons)?
20. Which counties’ centroid falls outside the polygon (label-placement risk)?
21. For [Ramsey County, MN], what is the area of the largest interior hole?
22. How many neighbors does [Utah County, UT] have?
23. Which county in [AL] has the most neighbors?
24. List counties in [CA] with exactly two neighbors.

All words between square brackets [] are just examples of parameter values and can be replaced by other values during the final test demonstration. For example, in prompt #1 (Show me [Madison County, Idaho]), “Madison County, Idaho” is a parameter value, and the same prompt can be posed as “Show me [King County, WA]”.

Project Key Elements and Milestones

Key Elements

- Frontend elements (FE): Map component, chat component, answers panel, and a log panel (shows NL-to-SQL outcomes, LM runtime, DB query time, etc).
- Language model (LM): interacts with the end user and the connectors to FE and DB.
- PostGIS database (DB): stores and queries USA county information.
- Connectors between all elements: FE to LM, LM to DB, DB to FE, etc.

Milestone 1

- Create a PostGIS database named “*USCountyDB*” with one table named “*counties*” that deploys the project dataset.

- Set up an LM and enable interaction with it for testing purposes. This can be deployed on your own device or a free service like Google Colab or another service. APIs to remote LLM are not allowed without a reasonable justification.

Milestone 2

- Build FE with preliminary functionality. It is accepted at this milestone that the connectors between FE and other key elements are not fully functioning.
- Build an NL-to-SQL connector between LM and DB. The connector should at least translate the NL prompt to a correct SQL statement, so we can pose it to the DB console and it works. The following paper and its related work could help you instruct the LM.
  Wang, Hui, Li Guo, Yubin Liang, Le Liu, and Jiajin Huang. "GPT-Based Text-to-SQL for Spatial Databases." ISPRS International Journal of Geo-Information 14, no. 8 (2025).

Milestone 3

- Build all fully functioning connectors between FE, LM, and DB.
- Fine-tune the LM to function accurately on all expected question templates.

Milestone 4

- Fix any remaining issues.
- Final test and demonstration.
- **Optional bonus:** enable your web application to call remote LLM APIs and provide a thoughtful comparison of performance and accuracy with the locally deployed LM. This comparison must be included in the final report with clear enough examples, performance numbers, and screenshots. Both local LM and remote LLM options must be available to be tested in the final demonstration.

Deliverables

Fall classes begin on Thursday, September 25, 2025, and end on Friday, December 5, 2025.

All deliverables are due at the class meeting on Mondays at 9:00 AM.

Week 1: Monday, September 29, 2025 (First class)

Week 2: Monday, October 6, 2025 (Group formation)

Week 4: Monday, October 20, 2025 (Milestone 1)

Week 6: Monday, November 3, 2025 (Milestone 2)

Week 8: Monday, November 17, 2025 (Milestone 3)

Week 10: Monday, December 1, 2025 (Milestone 4 and Final Report)

Grade Breakdown

	Points (%)	Evaluation Criteria
Milestone 1	15	Correctness of DB querying and availability of LM
Milestone 2	45	FE completeness, responsiveness, and elegance, NL-to-SQL accuracy
Milestone 3	15	Accuracy and performance
Milestone 4 and Final Report	20	All above
Attendance	5	Commitment to attend and work with your group

Web Application Interface

- The web application interface consists of four parts: a map component, a chat component, an answers panel, and a log panel.
- The map component must be interactive, meaning its zoom and pan views change with every new answer. For example, if the map currently shows a county in NY and the new prompt's answer shows counties in Arizona, the map must automatically clear the old shapes from NY, focus on Arizona, and show the new shapes.
- The polygons on the map component should be clickable and display the county information.
- The map component could use APIs of Google Maps, Bing Maps, ESRI Maps, OSM Maps, or any other mapping service that serves the purpose.
- The map component should be the largest possible in size compared to the whole screen.
- The chat component must enable the user to interact through **free text and NOT a predefined list of questions**. The sample questions above are just examples, not all the possible prompts. The prompts are in natural language, so any English sentence is valid to use.
- The chat component should show both user prompts and application responses, like any regular chatbot.
- It is your choice to integrate the answer panel within the chat component or make it separate. It is acceptable if the answer panel duplicates some of the chat box content if needed. It is all your design choice. The only requirement is to provide a decent user experience where all the history of prompts and their answers are clear and accessible.
- The log must show all the details and intermediate steps, including but not limited to NL-to-SQL outcomes, LM runtime, DB query time, etc. Log records of all history of prompts must be accessible and easy to see.

Important Notes

- Each project group is 3-4 students.
- Below is a short list of **computation resources** available for you. You are not limited to this list; these are just suggestions. Feel free to use your personal machine or any other resources if they are more helpful.
    - Upon forming your project group, the department technical staff will give your group shared access to a machine with a few GPUs of the model NVIDIA A6000 Ada, 48GB VRAM, full specs are available on https://www.nvidia.com/en-us/products/workstations/rtx-6000/. This will be shared access with other students, and each session will be limited to approximately 6 hours. Your job is submitted to a queue, and it runs after the current jobs finish.
    - Kaggle Notebooks provide free access to one NVIDIA Tesla P100 GPU, limited to approximately 30 hours per week. You can check online resources for more information.
      https://www.kaggle.com/docs/notebooks
      https://www.geeksforgeeks.org/machine-learning/how-to-use-gpu-in-kaggle/
    - The free version of Google Colab grants access to NVIDIA T4 GPUs, subject to quota restrictions and availability.
      https://colab.research.google.com/notebooks/pro.ipynb
      https://medium.com/data-science-in-your-pocket/understanding-google-colab-free-gpu-in-detail-15074081d494
    - Amazon SageMaker Studio Lab provides access to GPU compute up to 4 hours in a 24-hour period.
      https://docs.aws.amazon.com/sagemaker/latest/dg/studio-lab-overview.html
- The following are suggestions and pointers that might help, ***feel free to use them or replace them***:
    - Streamlit Framework provides chat elements for frontend interfaces
      https://docs.streamlit.io/develop/api-reference/chat
    - The LM could be:
        - **Phi-3-mini-4k-instruct** (3.8 B)
          https://huggingface.co/microsoft/Phi-3-mini-4k-instruct
          https://huggingface.co/microsoft/Phi-3-mini-4k-instruct-gguf
          https://huggingface.co/unsloth/Phi-3-mini-4k-instruct
        - **Gemma-2B-instruct** (2 B)
          https://huggingface.co/google/gemma-2b-it
          https://blog.google/technology/developers/gemma-open-models/
          https://cloud.google.com/vertex-ai/generative-ai/docs/open-models/use-gemma
    - Streamlit Framework provides chat elements for frontend interfaces

https://docs.streamlit.io/develop/api-reference/chat

- Deck.gl visualization framework handles several challenges for map-based visualization
  https://deck.gl/docs
- If needed, pgvector tool provides semantic search on top of PostgreSQL databases, the foundational DB for PostGIS
  https://github.com/pgvector/pgvector