# GenAI Assignment – Structured Data Question Answering Assistant

Objective

Build an AI-powered assistant that can answer natural language questions based on structured data provided as CSV files.

Your system should:  
- Interpret user questions  
- Identify the relevant dataset(s)  
- Generate and run appropriate SQL (or Pandas) queries  
- Return clear, human-readable answers

Provided Datasets

You will work with three CSV files:

* sample\_bo\_tbl\_large.csv (Daily usage & transactions)
* country — 3-letter code (AAA, BBB, etc.)
* date — Date of record
* total\_mins
* international\_mins
* sms
* total\_data\_usage
* payg\_amount
* sample\_sub\_details\_large.csv (Monthly subscriber stats)
* country
* channel
* date
* subs — Active subscribers
* netadds — Net additions
* churn — Churn count
* sample\_revenue\_large.csv (Monthly revenue)
* country
* channel
* date
* revenue
* net\_revenue

Example Queries Your Assistant Should Handle

* Basic
* What was the churn in AAA last month?
* Show total SMS sent by BBB this year.
* What was the average pay-as-you-go amount in CCC last quarter?
* Intermediate
* Compare net revenue across channels for DDD over the last 6 months.
* List countries with highest data usage in Q2.
* Show monthly net additions for AAA over the past year.
* Complex
* Which country had the highest churn-to-subscriber ratio in Q1?
* What’s the correlation between data usage and pay-as-you-go top-ups by country?
* Compare average net revenue of Online vs Retail channels in the last 3 months.
* Which countries had both above-average revenue and below-average churn last month?

Requirements

1️⃣ LLM Integration

- Use a public LLM (e.g., OpenAI, Claude, Mistral) or you can also use any model from Hugging Face.  
- The model should:  
 - Select relevant dataset(s)  
 - Formulate the query logic (SQL / Pandas)  
 - Optionally provide reasoning

2️⃣ Query Engine

- You may use Pandas, DuckDB, SQLite, or equivalent

- The generated query should run on the data

3️⃣ Interface

- CLI, web app, or notebook — up to you

- Should accept user queries in natural language

Bonus (Optional)

- Include reasoning / explanation in the response

- Show generated query to the user

- Add fallback handling for ambiguous or incomplete queries

Deliverables

- Code repository (e.g., GitHub)

- README.md with:  
 - Setup and run instructions  
 - Example queries and outputs  
 - Description of your prompting strategy

Evaluation Criteria

- Accuracy of answers

- Code quality and modularity

- Clarity of LLM prompting

- Usability of the solution