Natural Language Data Query Generator

PROJECT SYNOPSIS

OF MINOR PROJECT

7th Semester

BACHELOR OF TECHNOLOGY

(ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING)

SUBMITTED BY:-

Ritika chawla, 301310922036

Sahana Afreen, 301310922012

Shiv Prakash Singh,301310922051

Yukta verma,30131092206

RUNGTA COLLEGE OF ENGINEERING AND TECHNOLOGY

BHILAI (C.G)

Rungta College of Engineering and Technology, Bhilai
Department of CSE - AIML

Table of Contents

Guide Name

Prof. Pinki Patra

Consent of Guide

Suggestions by the guide

1. Introduction:

In today's data-driven world, extracting insights from datasets is a critical driver of innovation. However, this process remains a complex task, often requiring expertise in query languages like SQL or programming libraries like Pandas. This technical barrier poses a significant challenge for non-technical users. AI-driven query generation offers a transformative solution by leveraging Artificial Intelligence (AI) and Natural Language Processing (NLP) to translate plain English questions into executable data queries. This project explores the domain of "Natural Language Data Query Generation," utilizing cutting-edge AI to automate the data analysis process.

The consequences of this technical barrier are significant, creating an "analysis bottleneck" within organizations. Business analysts, marketing professionals, and key decision-makers often possess deep domain knowledge but lack the coding skills to query databases directly. This dependency on a small pool of technical experts leads to slow turnaround times for reports, stifles curiosity, and prevents the kind of iterative, exploratory data analysis that often uncovers the most valuable insights. The current paradigm forces a rigid, request-and-wait cycle, where the nuance of a business question can be lost in translation. Our project directly addresses this inefficiency by removing the intermediary, empowering domain experts to engage in a direct, conversational dialogue with their own data.

The core innovation of this project lies in its use of a state-of-the-art Large Language Model (LLM) as a dynamic "reasoning engine." Unlike older, rule-based systems that require extensive pre-programming for specific datasets, our approach is designed to be universally applicable. The true power of the LLM is its ability to understand not just the user's question, but also the unique context of the data they provide. By analyzing the schema—the column names, data types, and sample rows—of a user-uploaded file, the model constructs a tailored, executable query in real-time. This process goes far beyond simple keyword-to-command translation; it involves semantic understanding, logical inference, and the generation of precise, syntactically correct code.

The intersection of AI and NLP in data analysis aims to bridge the gap between human-readable questions and machine-executable code. AI-powered tools facilitate automatic query generation by analyzing text-based inputs and transforming them into optimized data retrieval commands. This project dives into the capabilities of a natural language query generator, assessing its impact on data analysis productivity, accessibility for non-technical users, and the overall accuracy of data retrieval. Ultimately, the vision is to create a tool that not only answers questions but also fosters a culture of data literacy, making data a truly democratized asset within any organization.

2. Rationale Behind the Study:

The motivation for this project is rooted in addressing several critical and persistent challenges in the field of data analytics. While the volume of available data has grown exponentially, the tools to access and interpret it have not kept pace in terms of accessibility for the average business user. This study is based on the rationale that a fundamental shift is needed—from complex, code-centric analysis to intuitive, conversation-driven exploration.

1. Accelerating Time-to-Insight: The platform aims to drastically shorten the data analysis cycle by automating query generation. In a traditional workflow, the journey from a business question to a data-driven answer is often long and inefficient, creating an "analysis bottleneck." A decision-maker first formulates a request, which is then passed to a technical data team, where it enters a queue. This process can take hours, or even days, during which time the business context may change or the opportunity for a timely decision may be lost. For example, a sales manager noticing a sudden dip in regional performance would have to wait for a formal report, by which time a competitor might have solidified its advantage. By translating plain English questions into executable code instantly, our system collapses this protracted timeline into seconds. This is not merely a time-saving measure; it represents a paradigm shift from reactive reporting to proactive exploration. It enables users to move from static, periodic reports to a dynamic, real-time analytical process, allowing them to focus their cognitive energy on interpreting insights and formulating strategy, rather than on the mechanics of data retrieval.

2. Democratizing Data Access: By removing the need for specialized programming knowledge, the platform empowers non-technical users such as business analysts, managers, and domain experts. The term "data democratization" means putting the power of data directly into the hands of those who have the most context to understand it, breaking down the organizational silos where data is often held captive by a few technical gatekeepers. A marketing manager should not need to know SQL to check the performance of a recent campaign across different regions. Similarly, a finance professional should be able to ask about budget variances without complex spreadsheet manipulations, and a supply chain manager can query inventory levels without a technical intermediary. Our system provides these professionals with direct, intuitive access to their data, which leads to better, more context-aware questions. This fosters a more agile, data-driven culture where anyone can make informed decisions based on evidence, not just intuition.

3. Literature Review

S.No.	Author's Name	Title	Source	Year	Methodology	Findings	Gaps
1.	Lei et al.	SPIDER 2.0: Evaluating language models on real-world enterprises text-to SQL workflows	ICLR	2025	Introduced a new, highly complex benchmark (Spider 2.0) using massive, real-world enterprise databases to evaluate how well modern AI models handle true business complexity.	Found that even the most advanced LLMs perform poorly on this realistic benchmark, highlighting a Significant gap between academic performance and real-world enterprise needs.	This paper's primary contribution is evaluation and problem identification, not a new solution. It demonstrates the need for a more robust system but does not build one.
2.	Sun et al.	SQL-PaLM: Improved large language model adaptation for Text-to- SQL	arXiv	2024	Proposed a comprehensive framework to adapt a large language model (PaLM-2) for Text- to-SQL using advanced fine-tuning, synthetic data, and context-aware schema selection.	Showed that a holistic approach with fine-tuning and rich contextual information yields substantial accuracy improvements on major Text-to-SQL benchmarks like Spider and BIRD.	The focus is on achieving state-of-the-art accuracy on existing benchmarks. It does not address the specific challenge of an end-user application that handles arbitrary, user-uploaded data files.
3.	Franciscatto et al.	Talk to Your Data: Chatbot System f or Multidimens ional Datasets	IEEE COMPS AC	2022	Developed an intent- based chatbot that reads a predefined database schema to guide users through a structured conversation, building a multidimensional query step-by-step.	Demonstrated that a conversational interface can successfully empower non-technical users to query complex data, finding the approach useful and faster than manual searching.	Relies on a fixed, known database schema and cannot handle user-uploaded datasets. The conversational flow is rigid and not suited for truly open-ended, flexible user questions.

4.	Elgohary et al.	Speak to your Parser: Interactive Text-to-SQL with Natural Language Feedback	ACL	2020	Created a system where users can correct an inaccurate SQL query by providing free-form natural language feedback, which the model then uses to generate a corrected parse.	Proved that interactive natural language feedback can significantly improve overall accuracy, but also highlighted that the task is very challenging for models of that era.	The study focuses on *correcting* existing queries, not generating them from scratch for a user's own data. The technology predates modern, more powerful LLMs.

4. Objectives

Based on the gaps identified in current research, this project aims to achieve the following objectives:

1. **(Paper I & II)-** Architect an accessible, data-agnostic analysis system capable of ingesting and processing user-uploaded datasets (e.g., CSV files). This directly addresses the critical schema-dependency gap identified in foundational works.
2. **(Paper III)-** Empirically validate the system's query generation accuracy through a controlled evaluation, with the explicit goal of establishing a high-performance benchmark (targeting a 90% success rate) for arbitrary user-provided data.
3. **(Paper IV)-** Conduct a comparative analysis to quantify the performance gains of a modern LLM-based architecture against a non-grounded baseline model. This will demonstrate the significant improvements in query generation flexibility and accuracy afforded by current generative AI.

5. Feasibility Study:

Technical Feasibility:

The system relies on mature technologies like Python/Django and the Pandas library, plus an external Generative AI API. Key challenges include handling diverse CSV data structures uploaded by users, prompt engineering, and ensuring the safe execution of generated code.

Economic Feasibility:

Cost Analysis: Initial costs involve web application development and UI design. Ongoing expenses include API usage fees (pay-per-call), cloud hosting for the web server, and personnel costs for system maintenance and future updates.

Environment Feasibility:

Impact Assessment: The project's AI processing requires significant computing power, but this environmental load is outsourced to the large-scale, optimized data centers of the LLM provider, minimizing our direct energy consumption.

6. Timeline and Implementation Plan:

Timeline: Establish a structured timeline covering UI design, backend development, AI API integration, testing, and deployment, ensuring smooth progression through project milestones.

Implementation Plan: Define roles, allocate resources, and outline budgeting requirements. Develop monitoring mechanisms to track progress and ensure timely completion, optimizing efficiency and system performance.

7. Methodology:

7.1 Data Ingestion and Initial Processing:

The process begins when the user provides a CSV file and a natural language question. The system ingests these inputs and immediately loads the data into a Pandas DataFrame, which serves as the primary structure for all analysis.

7.2 Schema Extraction and Context-Aware Prompt Engineering:

Once the data is loaded, the system performs an automated schema analysis to extract all column names and their inferred data types. This structural metadata is then combined with the user's question to construct a detailed, context-aware prompt for the AI model.

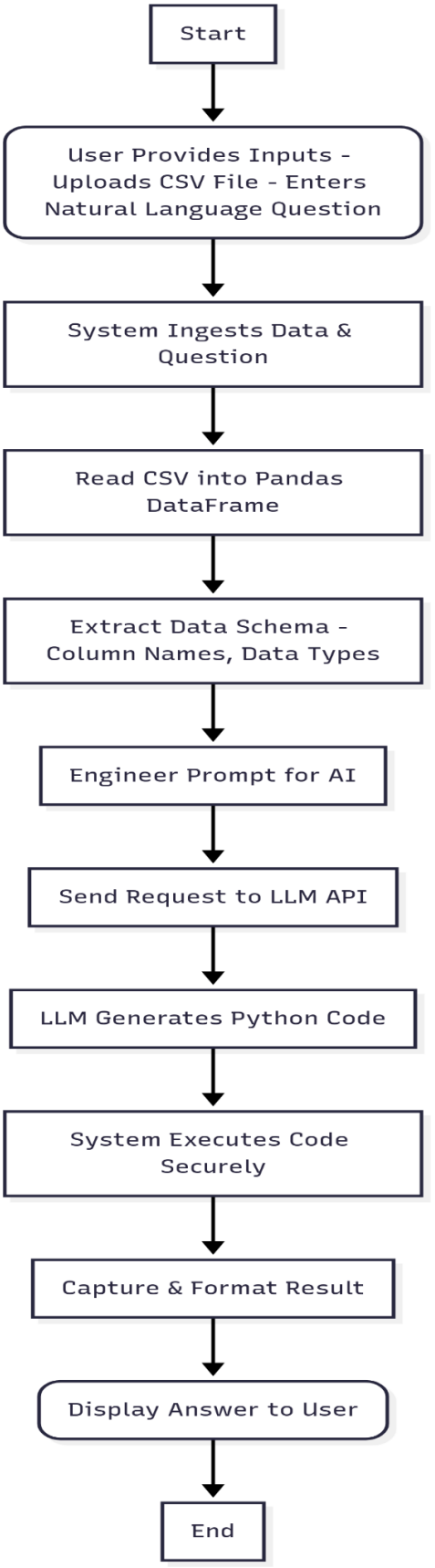
7.3 AI-Powered Code Generation via LLM:

The engineered prompt is sent as an API request to a Large Language Model (LLM). The LLM processes this contextual prompt and generates a Python code snippet, specifically for the Pandas library, to perform the requested analysis.

7.4 Secure Execution and Result Presentation:

The AI-generated Python code is run within a secure, sandboxed environment to produce a result. This raw output is then captured, formatted into a clean, human-readable presentation, and displayed back to the user as the final answer.

8. FLOW CHART:


Start
User Provides Inputs - Uploads CSV File - Enters Natural Language Question
System Ingests Data & Question
Read CSV into Pandas DataFrame
Extract Data Schema - Column Names, Data Types
Engineer Prompt for AI
Send Request to LLM API
LLM Generates Python Code
System Executes Code Securely
Capture & Format Result
Display Answer to User
End

9. Natural Language to Code Generation process:

- **Input Ingestion:** Gather the user's inputs, which include the uploaded CSV dataset and the natural language question about that data.
- **Context Extraction:** Analyze the uploaded data to dynamically understand its schema, including column names, data types, and sample rows for context.
- **Prompt Engineering:** Construct a detailed prompt for the AI, combining the extracted data schema, user question, and specific instructions to generate Python Pandas code.
- **Code Generation:** Generate an executable Python code snippet based on the AI-processed prompt, considering the specific structure of the user's data.
- **Secure Execution:** Safely run the dynamically generated code in a controlled environment against the in-memory data from the user's file.
- **Final Answer Output:** Produce the final, human-readable answer by capturing the output of the executed code and presenting it back to the user.

10. Facilities required:

- **Data Ingestion System -** A web interface to allow users to upload and process their CSV data files securely and efficiently.
- **Generative AI Service -** An API-based connection to a powerful Large Language Model like Google's Gemini for generating the data query code.
- **In-Memory Data Handling -** A robust backend system for temporarily storing and managing the user-uploaded data in a Pandas Data Frame for analysis.
- **User-Facing Web Interface -** A user-friendly web application for submitting natural language questions and clearly visualizing the final textual answers.
- **Error Handling and Execution System -** Automated tools for safely executing the generated code and managing any potential errors gracefully.
- **API Integration Capabilities -** A secure and reliable API client for connecting with the external Large Language Model service.
- **Scalable Cloud Infrastructure -** Cloud-based resources to ensure the web application is scalable, accessible, and can efficiently handle user requests.

11. Expected outcomes:

- **Accurate Query Generation:** A machine learning model that accurately translates natural language questions into executable code based on the context of a user's dataset.
- **On-Demand Data Analysis:** A real-time, web-based tool allowing non-technical users to get immediate answers from their data without writing any code.
- **Increased Data Accessibility:** A final application that successfully empowers users to explore their datasets conversationally, leading to faster and more widespread data-driven decision-making trends.