



## Enhancing Database Interaction with Large Language Model

<sup>1</sup>Dr.Mallikarjun C Sarsamba, <sup>2</sup>Prof. Govind M R, <sup>3</sup>Dr.Chinna V Gowdar

<sup>1</sup>Professor,ECE Dept, Hirasugar Institute of Technology ,Nidasoshi

<sup>2</sup>Assistant Professor,Dept.of ECE,Jain College of Engineering and Research,Belagavi

<sup>3</sup>Associate Professor, Dept.of ECE, RYMEC,Bellary

### ABSTRACT

Many users struggle with SQL querying due to its complex syntax and structured format. Querying a database traditionally requires a deep understanding of data distribution and internal dependencies, making it challenging for non-experts. This project simplifies database querying by developing an application that allows users to input queries in natural language, eliminating the need for SQL expertise. By leveraging the Gemini API for natural language processing, the application translates user queries into SQL, enabling effortless data retrieval. This approach integrates natural language processing and generative AI, making database interactions more accessible to a broader audience. Additionally, the application offers automated data analysis and visualization features, helping users gain insights regardless of their technical proficiency. Traditional SQL methods often require specialized knowledge, restricting access to those with technical expertise. By automating query construction and providing intuitive data visualizations, this solution empowers users to explore and understand their data more effectively. The software stack includes Python for backend processing, SQLite for database management, and a web-based frontend for seamless user interaction, delivering a comprehensive and user-friendly approach to database querying and analysis.

**Keywords**—Generative AI, Natural Language Processing, Text to SQL Generation, Gemini API, Large Language Model, Machine Learning, Deep Learning, Google LLM, SQLQuery Generation, Prompt Engineering, Data Retrieval

### I. INTRODUCTION

Proficiency in SQL (Structured Query Language) is essential for retrieving and analyzing data from databases. However, many users find SQL challenging due to its complex syntax and structured format, requiring specialized knowledge and training. This limitation restricts database access to individuals with technical expertise, hindering efficient data analysis and decision-making. To address this challenge, our study leverages generative AI and natural language processing (NLP) to introduce a more intuitive approach to database querying. The goal is to develop a system that allows users to input queries in natural language instead of SQL, making data retrieval and analysis more accessible to a broader audience.

Generative AI will generate accurate SQL queries, enabling the application to retrieve the required data from the database seamlessly. By integrating natural language processing (NLP) and generative AI, this project transforms the way databases are accessed, allowing users to interact with them using natural language instead



of complex SQL syntax. The primary objective of this application is to make database querying more accessible to non-technical users by abstracting SQL queries into a simpler and more intuitive format. Additionally, the application will include visualization tools for data analysis, enhancing user experience by providing clear insights. It will also have the capability to automatically analyze database content and generate relevant insights based on data descriptions, helping users make informed decisions. This project aims to develop a simple, user-friendly, and efficient application that leverages NLP and generative AI to streamline database querying. By adopting this innovative approach, data analysis becomes more accessible and intuitive, empowering users to extract meaningful insights without requiring SQL expertise.

## II. LITERATURE SURVEY

Ahkouket et al. [1] A hypothetical scenario has been presented to illustrate the translation of natural language queries into SQL queries, aligning with the goals of this research. Developing a system that enhances querying requires proficiency in both SQL and Natural Language Processing (NLP) tools. Furthermore, the existing research by Baig et al. [2] plays a crucial role in exploring recent advancements and challenges in translating natural language into SQL. Their study provides valuable insights into the progress made in this field and the obstacles that remain. A descriptive analysis of these advancements in natural language querying is essential for the context of this study, as it helps identify key developments and areas for improvement in making database interactions more intuitive and accessible.

Berti et al. Ferreira et al. [3] Large Language Models (LLMs) have been applied to process mining, marking an important step in evaluating their utility and challenges in complex data processing. This study highlights the flexibility of LLMs in handling various aspects of data processing, offering valuable insights into improving database query processing and analysis within the application. In this context, Bukhari et al. [4] conducted a literature review on NLP-to-database (NLP-to-DB) querying frameworks, examining approaches for querying databases using natural language. Their findings on designing and developing user-friendly database interfaces are highly relevant to this research. The lessons learned from their study can be applied to the development of an application aimed at simplifying the querying process, making it more intuitive and accessible for users.

Fang et al. [5] A survey was conducted on the use of Large Language Models (LLMs) in tabular data analysis to assess their advantages and limitations in handling structured data. The study concluded that LLMs enhance an application's ability to process and understand SQL queries over structured databases. To improve text-to-SQL translation and ensure the success of our project, we incorporate techniques from Hui et al. and schema dependency analysis from Gilbert et al. [6]. Their methods contribute to more efficient query development by accurately converting natural language questions into SQL queries. By leveraging these advancements, the project aims to streamline the querying process, making it faster, more accurate, and user-friendly.

Liu et al. [7] Research efforts have focused on enhancing LLM performance on relational data by optimizing worst-case LLM queries in relational workloads. These improvements contribute to faster and more accurate query generation, ultimately enhancing the application's efficiency, end results, and overall user experience. Additionally, Nethravathi et al. [8] explored techniques for translating user-input natural language queries into structured query language, with a focus on understanding the semantics of query interpretation. Their work offers key insights into how an application's structure influences the challenges and solutions in transforming



natural language into SQL queries. By incorporating these findings, this project aims to develop a robust and intuitive system for seamless database querying.

Shen, Yi et al. [9] A review of the literature on natural language interfaces (NLIs) for data visualization was conducted to understand how NLIs contribute to developing natural language comprehension for data interaction. NLIs enable users to interact with visualizations through natural language commands, making data exploration more intuitive. Insights from this survey can inform the design and implementation of how natural language processing (NLP) integrates with databases in this application. Sturley et al. examined the role of generative AI in SQL generation, while Zoph et al. [10] were among the first to introduce a generative AI architecture leveraging EBNF context-free grammars to automate SQL statement generation. Their approach aligns with this project's objectives by utilizing EBNF grammar-based automation, leading to more efficient SQL generation with fewer errors. By adopting their methodology, this application aims to streamline database querying, reducing complexity and improving accuracy.

Uddin et al. projected a scalable NoSQL query generation technique for natural language question recommended that BERT models such as [11] can effectively convert natural language question to NoSQL queries by employing deep learning mechanism. Their main work is done in the field of increasing speed and accuracy in order to translate questions from plain language to NoSQL, by means of advanced deep learning techniques. However, the BERT approach was working quite well for natural language processing. The idea is to use this style in the design of NoSQL queries, to especially better efficiency dealing with complex database structures. Zhang et al. [12] obtainable a thorough examination for comparing the text-to-SQL capabilities of LLMs. Their work sheds light on the pros and cons of LLMs for generating SQL queries from natural language.

### III. PROPOSED WORK

The entire process that was recommended without using SQL queries how to access the data and retrieve the related information. The following phases describe the approach used in this work.

#### A. *Natural Language Processing (NLP):*

Every request made by the user moves up from the Natural Language Processing (NLP) Module. This module performs all the needed preprocessing tasks like tokenization, stemming, and POS tagging using libraries like NLTK and spaCy. These processes extract important data from the user inputs so that various analyses could be performed. This module is based on the Google Generative AI API that provides comprehensive language understanding capabilities. This integration allows to the system to use its analysis capabilities to infer the important data in original user queries and perform a semantic analysis that reads and understands the Natural Language Inputs more easily.

#### B. *Generative AI for SQL Query Generation:*

The Generative AI module is essential for converting natural language questions into SQL queries. It leverages generative AI techniques and the Gemini LLM API to generate accurate SQL queries based on **user** intent and meaning. To enhance precision, the module is designed to consider user-defined key values, enabling context-aware query generation that aligns with the database schema. By integrating generative AI, the system significantly improves its ability to interpret user requests accurately, ensuring more efficient and seamless database interactions. As a result, query processing becomes faster, more accurate, and better optimized,

enhancing overall system performance and usability.

### C. Database Interaction:

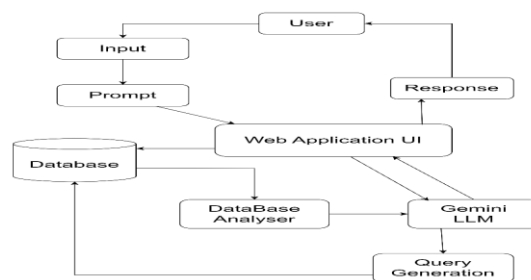
The Database Interaction module is employed to facilitate the interactions of the application with the underlying SQLite database. This is the Python module is used by the sqlite3 package to run SQL queries on the database and to handle the results of the queries. It also incorporates healthy error handling mechanisms, derived to manage even the most intricate or slippery questions to attain compliance with success and retrieval of answers from a query. In addition to this, the module also supports features like data fetching, modification, and removal which in turn assist in the in-depth interaction with the database from inside the application itself.

### D. Web Application Frontend:

The Web Application Frontend is an interface with which users interact with the system. Built with Streamlit, This UI provides user-friendly access to database information. This system includes an interactive input interface for users to input their queries, shows query results in a human-readable way, and supports interaction with visualization of data. The frontend is uniquely designed to enhance the user interactions thereby making the database querying process available and easy to even people who are not technically competent. Its front-end makes it easy to examine and view the data, with a great look-and-feel; it is super interactive and well designed.

### E. Automated Data Analysis and Visualization:

The Automated Data Analysis and Visualization module extends the Hydra hammer's capabilities by analyzing the query results and rendering insightful visualizations. The library that we are going to use to do the data analysis on that query result includes Pandas, Matplotlib. It monitors behavior (patterns) and outputs visualizations like graphs and charts. These visualizations give consumers data-relevant insights, and the data-generated information or findings becomes an easier phenomenon in data discovery making it more prone to decision analysis. Using this module not only makes user easily understand but also helps in automated analysis as well as visualization which our data analyst can lead to data driven decision-making.



**Figure 1. Architecture for SQL Query Generation**

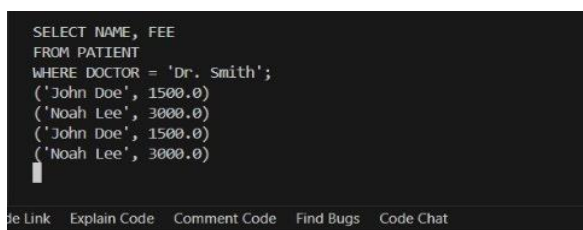
The figure 1 shows the architecture diagram for SQL query generation using Gemini LLM. The initial process begins when the user inputs natural language query to access the database. The next process is to input the user input along with schema of the database to the Gemini LLM where it converts this natural Language query into SQL query. Finally the required information will be retrieved and displayed to the user.

## IV. EXPERIMENTAL RESULTS

The testing conducted for the overall evaluation of the natural query interface (QOD framework) built in this project is represented in Figures 2 and 3. The system was thoroughly tested and analysed to assess its response

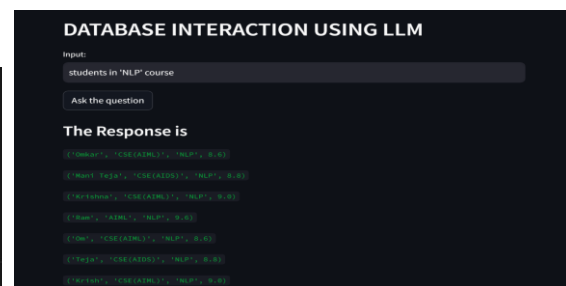


quality, focusing on query comprehension, processing efficiency, and accuracy in converting natural language into SQL. Key evaluation criteria included the system's ability to retrieve information from the database, its responsiveness, and the accuracy of query translation. The results indicate that the system effectively processes various types of natural language inputs, demonstrating its capability to handle diverse user inquiries. Additionally, the system exhibited minimal time lag in query execution, ensuring quick access to relevant data. Overall, the experimental results confirm the effectiveness and efficiency of the natural language database interface, highlighting its potential to simplify database querying while maintaining high performance and accuracy.



```
SELECT NAME, FEE
FROM PATIENT
WHERE DOCTOR = 'Dr. Smith';
('John Doe', 1500.0)
('Noah Lee', 3000.0)
('John Doe', 1500.0)
('Noah Lee', 3000.0)
```

**Figure 2. Generating SQL Query Using Gemini LLM**



**DATABASE INTERACTION USING LLM**

Input:  
students in "NLP" course

Ask the question

**The Response is**

```
SELECT NAME, ID FROM STUDENT WHERE COURSE = 'NLP';
('John Doe', 1)
('Noah Lee', 2)
('John Doe', 3)
('Noah Lee', 4)
('John Doe', 5)
('Noah Lee', 6)
('John Doe', 7)
('Noah Lee', 8)
('John Doe', 9)
('Noah Lee', 10)
```

**Figure 3. Using Natural Language to Query Database**

Input is given in the prompt box it was searched and provided the related all output in the green color text without using the SQL comment.

## V. CONCLUSION

The paper introduces an innovative approach to database querying using the Gemini API and a natural language interface powered by Large Language Models (LLMs). This Natural Language Querying method allows users to interact with databases using human-like language instead of SQL, making database access more intuitive and accessible to non-technical users. By integrating generative AI, particularly the Gemini LLM, the system enhances user experience, streamlines complex database operations, and simplifies data retrieval and analysis. The querying process leverages the Gemini API, which is set up using a Google API key. Users input queries in plain English, and the Gemini LLM translates them into SQL queries through a user-friendly Streamlit web app. These queries interact with an SQLite database, and the results are displayed in an easily interpretable format. The combination of natural language querying and generative AI has the potential to revolutionize fields like data science, business intelligence (BI), **and research** by making database interactions more efficient and user-friendly. Automated data analysis within the system facilitates simplified querying, enhanced data exploration, and improved decision-making. Ultimately, this project democratizes database access, empowering users of all skill levels to query and analyze data effortlessly.

## REFERENCES

- [1] Ahkhouk, Karam, and Mustapha Machkour. "Towards an interface for translating natural language questions to SQL: a conceptual framework from a systematic review." *International Journal of Reasoning-based Intelligent Systems* 12, no. 4 (2020): 264-275.





- [2] Baig, Muhammad Shahzaib, Azhar Imran, Aman Ullah Yasin, Abdul Haleem Butt, and Muhammad Imran Khan. "Natural language to sql queries: A review." *Technology* 4, no. 1 (2022): 147-162.
- [3] Berti, Alessandro, Humam Kourani, Hannes Hafke, Chiao-Yun Li, and Daniel Schuster. "Evaluating Large Language Models in Process Mining: Capabilities, Benchmarks, Evaluation Strategies, and Future Challenges." *arXiv preprint arXiv:2403.06749* (2024).
- [4] Bukhari, Syed Ahmad Chan, Hafsa Shareef Dar, M. Ikramullah Lali, Fazel Keshtkar, Khalid Mahmood Malik, and Seifedine Kadry. "Frameworks for querying databases using natural language: A literature review–NLP-to-DB querying frameworks." *International Journal of Data Warehousing and Mining (IJDWM)* 17, no. 2 (2021): 21-38.
- [5] Fang, Xi, Weijie Xu, Fiona Anting Tan, Jiani Zhang, Ziqing Hu, Yanjun Qi, Scott Nickleach, Diego Socolinsky, Srinivasan Sengamedu, and Christos Faloutsos. "Large Language Models on Tabular Data--A Survey." *arXiv preprint arXiv:2402.17944* (2024).
- [6] Hui, Binyuan, Xiang Shi, Ruiying Geng, Binhua Li, Yongbin Li, Jian Sun, and Xiaodan Zhu. "Improving text-to-sql with schema dependency learning." *arXiv preprint arXiv:2103.04399* (2021).
- [7] Liu, Shu, Asim Biswal, Audrey Cheng, Xiangxi Mo, Shiyi Cao, Joseph E. Gonzalez, Ion Stoica, and Matei Zaharia. "Optimizing LLM Queries in Relational Workloads." *arXiv preprint arXiv:2403.05821* (2024).
- [8] Nethravathi, B., G. Amitha, Anusha Saruka, T. P. Bharath, and Setu Suyagya. "Structuring natural language to query language: a review." *Engineering, Technology & Applied Science Research* 10, no. 6 (2020): 6521-6525.
- [9] Shen, Leixian, Enya Shen, Yuyu Luo, Xiaocong Yang, Xuming Hu, Xiongshuai Zhang, Zhiwei Tai, and Jianmin Wang. "Towards natural language interfaces for data visualization: A survey." *IEEE transactions on visualization and computer graphics* 29, no. 6 (2022): 3121-3144.
- [10] Troy, S. Sturley, J. M. Alcaraz-Calero and Q. Wang, "Enabling Generative AI to Produce SQL Statements: A Framework for the Auto- Generation of Knowledge Based on EBNF Context-Free Grammars," in *IEEE Access*, vol. 11, pp. 123543-123564, 2023.
- [11] Uddin Mohammed Nasir, Hossen, Kazi Mojammel, Minhazul Arefin, and Md Ashraf Uddin. "Bert model-based natural language to nosql query conversion using deep learning approach." *International Journal of Advanced Computer Science and Applications* 14, no. 2 (2023).
- [12] Zhang, Bin, Yuxiao Ye, Guoqing Du, Xiaoru Hu, Zhishuai Li, Sun Yang, Chi Harold Liu, Rui Zhao, Ziyue Li, and Hangyu Mao. "Benchmarking the Text-to-SQL Capability of Large Language Models: A Comprehensive Evaluation." *arXiv preprint arXiv:2403.02951* (2024).