

# Natural Language Interface for RDB using LLMs

**Ms.R.Femimol, Tharikoppula Triveni, Thatha Keerthi, Thummalapalli Harsha, Thupili Benhar Asish**  
*ASST.Professor(CSE), UG Scholar, UG Scholar, UG Scholar, UG Scholar*  
*Department of Computer Science & Engineering*  
*Bharath Institute of Science and Technology, BIHER*  
173,Agaram Road , Selaiyur, Tambaram, Chennai, Tamil Nadu, India

**Abstract :** In today's data-driven world, large volumes of structured data are stored in relational databases, which typically require knowledge of Structured Query Language (SQL) for data retrieval, creating a barrier for non-technical users. This project presents a Natural Language Interface for Relational Databases using Large Language Models (LLMs), aimed at simplifying database interaction by allowing users to query data using natural language. The system is implemented using a Streamlit-based chatbot interface where users can input queries in simple English. The system interprets user intent and generates the corresponding SQL query based on the database schema, which is then executed on a MySQL Employees database to retrieve relevant information. The results are further converted into meaningful natural language responses and displayed to the user along with the generated SQL query to improve transparency. The proposed system supports efficient handling of queries involving filtering, aggregation, and table relationships. It enhances accessibility by reducing dependency on technical expertise and provides an intuitive platform for database interaction. Overall, the project demonstrates how LLM-based approaches can effectively bridge the gap between non technical users and relational database systems.

**IndexTerms - Text-to-SQL, Natural Language Processing, Large Language Models, Database Querying, MySQL, Streamlit, Natural Language Interface to Database**

## I. INTRODUCTION

In modern organizations, large amounts of structured data are stored in relational databases. Accessing and analyzing this information typically requires the use of Structured Query Language (SQL), which allows users to retrieve, filter, and manipulate data. However, writing SQL queries requires technical knowledge of database schemas, syntax rules, and query structures. This creates a significant challenge for non-technical users who need to access information stored in databases but lack the required programming or database skills.

Recent advancements in natural language processing and large language models have enabled new approaches for simplifying human-computer interaction. Natural language interfaces allow users to interact with computer systems using everyday language rather than complex programming commands. In the context of databases, Text-to-SQL systems convert user questions written in natural language into SQL queries that can be executed on relational databases.

This project proposes an intelligent Text-to-SQL chatbot that allows non-technical users to retrieve data from a database using simple English queries. The system provides a user-friendly interface built with Streamlit, where users can enter their questions. A large language model interprets the user query and generates the corresponding SQL command. The generated query is executed on a connected database, and the results are displayed in an easy-to-understand format. By integrating natural language processing with database querying, the proposed system aims to reduce the dependency on technical experts and improve accessibility to data. This approach enables faster decision-making and simplifies the process of extracting meaningful insights from structured databases.

Furthermore, the proposed system emphasizes usability and simplicity, ensuring that users can interact with databases without requiring prior technical training. By providing an intuitive interface and automated query generation, the system minimizes the learning curve and enhances user experience. This is particularly beneficial in environments where quick access to information is critical for operational efficiency. In addition, the integration of large language models enables the system to handle a wide range of user queries, including complex and context-based questions. The system can interpret user intent more effectively compared to traditional rule-based approaches, thereby improving the accuracy and relevance of the generated SQL queries. This adaptability makes the system suitable for various application domains such as education, business analytics, and data management systems.

Moreover, the system can be extended with additional features such as query validation, error handling, and result visualization to further enhance its performance and reliability. These enhancements ensure that the system not only generates correct SQL queries but also presents results in a meaningful and interpretable manner. As a result, the proposed approach contributes to bridging the gap between complex database technologies and non technical users, making data access more efficient and inclusive.

## II. LITERATURE REVIEW

Several research studies have explored methods for enabling users to interact with databases using natural language instead of traditional SQL queries. These studies focus on improving accessibility, accuracy, and efficiency in retrieving information from relational databases. Majhadi Khadija and Machkour Mustapha proposed the CHAT-SQL system, which uses deep learning techniques to convert natural language queries into SQL statements. Their approach is based on an encoder–decoder architecture using Long Short-Term Memory (LSTM) networks. The model learns relationships between user questions and database structures to generate SQL queries automatically. Although the system demonstrates promising results, it still faces limitations when handling complex queries and large database schemas.

Arnav Jha and colleagues presented a system for converting natural language text into SQL queries using generative artificial intelligence techniques. Their research highlights the potential of transformer-based models to understand user intent and generate syntactically correct SQL queries. The study shows that generative models can improve the accuracy of query generation; however, the approach mainly focuses on query generation rather than providing a complete end-to-end user interaction system.

Xiaohu Zhu and co-authors conducted a comprehensive survey on large language model-based Text-to-SQL systems. The study categorizes different approaches such as prompt-based methods, fine-tuned models, and agent based architectures. The survey concludes that large language models significantly improve the ability of systems to understand complex natural language queries. However, effective integration with database systems and query validation remains an important challenge.

Another study on end-to-end Text-to-SQL generation within an analytics insight engine focuses on building a complete pipeline that integrates schema linking, query generation, and validation mechanisms. The system aims to improve reliability by incorporating knowledge about database schemas and automatically correcting errors in generated queries. While the framework provides a robust architecture, it is designed primarily for enterprise analytics environments and may be complex for smaller applications.

Recent research on LLM-powered SQL querying systems demonstrates how large language models can transform natural language questions into executable database queries. These systems combine semantic understanding with database execution mechanisms to retrieve accurate results. The research highlights the importance of combining language models with validation and execution components to ensure reliable query processing for non technical users.

From the analysis of these studies, it can be observed that although significant progress has been made in Text-to-SQL technologies, there is still a need for user-friendly systems that integrate natural language understanding, query generation, and real database execution. The proposed project addresses this gap by developing a Streamlit-based interface that allows non-technical users to interact with databases through natural language queries.

Furthermore, many existing approaches emphasize model performance and accuracy but often overlook usability and system accessibility for end users. In real-world scenarios, non-technical users require intuitive interfaces that not only generate SQL queries but also present results in a clear and interpretable format. Therefore, the integration of user friendly visualization tools and interactive interfaces becomes equally important as the underlying model performance.

In addition, challenges such as handling ambiguous queries, domain-specific database schemas, and ensuring data security continue to impact the effectiveness of Text-to SQL systems. Robust query validation, error handling, and controlled database access are essential to prevent incorrect or harmful queries. Addressing these challenges is necessary to make such systems reliable for practical deployment in various domains such as education, healthcare, and business analytics.

Finally, the rapid advancements in Large Language Models provide new opportunities to enhance Text-to-SQL systems by improving contextual understanding and adaptability across different domains. By combining LLM capabilities with efficient database interaction and intuitive user interfaces, modern systems can bridge the gap between complex database technologies and non-technical users, making data access more inclusive and efficient.

### III. METHODOLOGY

The proposed system follows a structured process to convert natural language queries into SQL queries and retrieve data from the database.

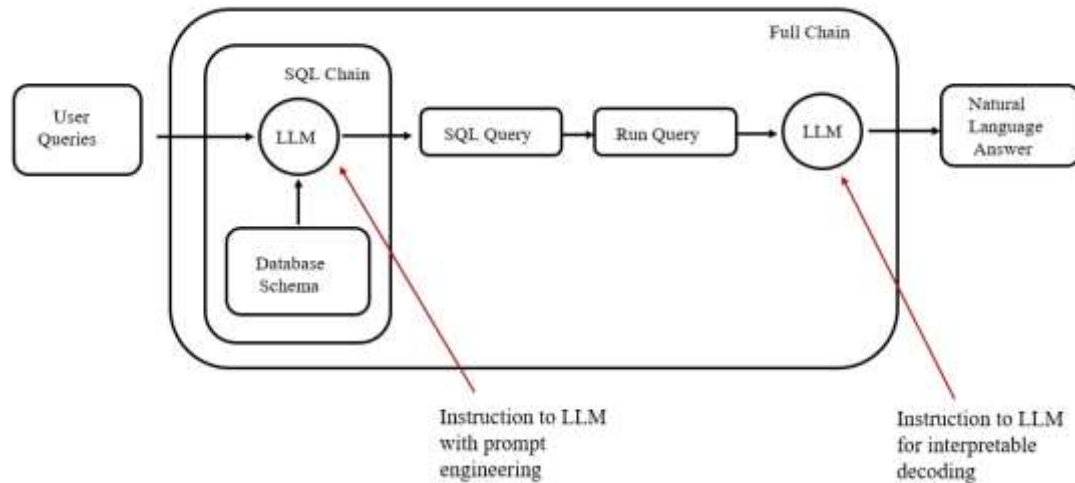


FIG. 1. System architecture Diagram

The working process consists of the following steps:

#### Step 1: User Query Input

The process begins when the user enters a query in natural language through the Streamlit-based user interface. The system is designed for non-technical users; therefore, no prior knowledge of SQL or database schema is required. The interface provides a simple and interactive environment for user input.

#### Step 2: Query Processing

Once the query is received, the system performs preprocessing operations such as text normalization, removal of unnecessary characters, and formatting. This step ensures that the input is structured properly before being passed to the language model for interpretation.

#### Step 3: Natural Language Understanding

In this step, a Large Language Model (LLM) analyzes the processed input to understand the user's intent. The model interprets the semantics of the query and identifies key components such as relevant tables, attributes, conditions, and relationships within the database schema.

#### Step 4: SQL Query Generation

Based on the interpreted information, the system generates an appropriate SQL query using prompt-based techniques. The generated query accurately reflects the user's request and ensures proper syntax and structure for execution on the database.

#### Step 5: SQL Validation (Optional but Important Enhancement)

Before execution, the generated SQL query can be validated to ensure correctness, safety, and compatibility with the database. This step helps prevent errors such as invalid syntax or unintended operations.

#### Step 6: Database Execution

The validated SQL query is executed on the connected MySQL database. The database engine processes the query and retrieves the relevant data based on the specified conditions.

#### Step 7: Result Retrieval

After execution, the database returns the result set to the application. The retrieved data may include multiple records depending on the query.

**Step 8: Result Display**

Finally, the system presents the retrieved results through the Streamlit interface in a structured and user-friendly format, such as tables or lists. This ensures that the output is easy to understand and interpret for non-technical users.

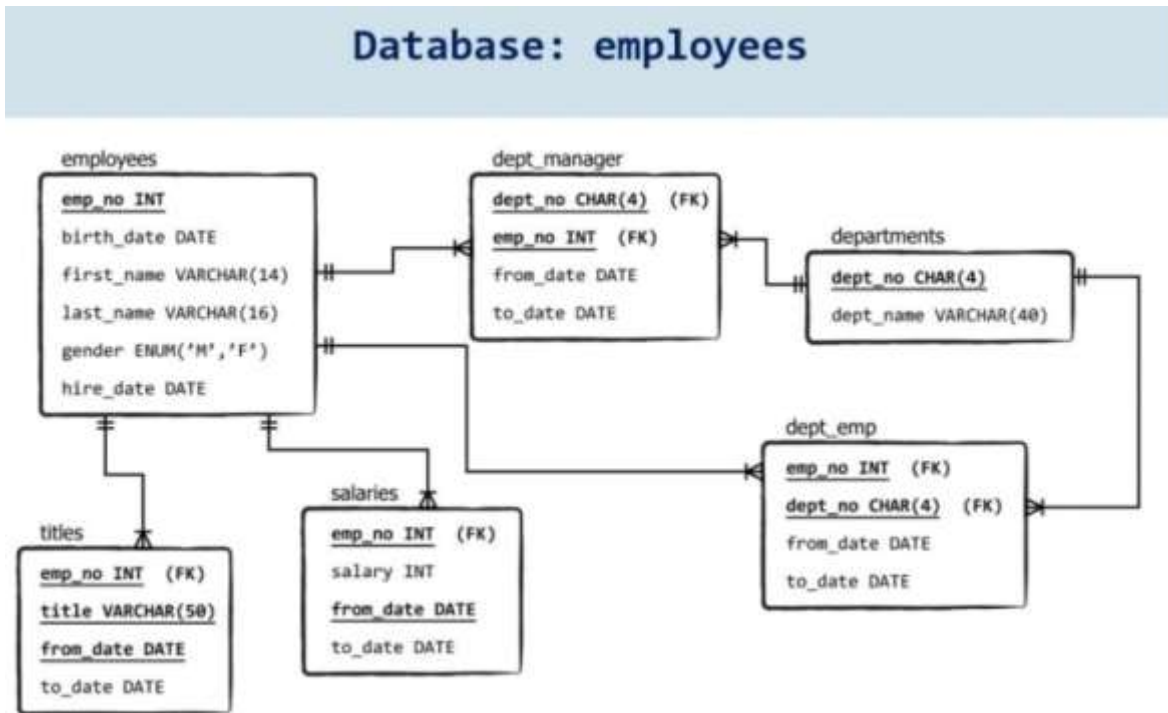


FIG.2. Database ER diagram

**IV.RESULT**

The proposed Text-to-SQL system was implemented to allow users to retrieve database information using natural language queries. The system was developed using a Streamlit interface, a Large Language Model for query interpretation, and a MySQL database for storing and retrieving data. After integrating these components, the system was tested with different user queries to evaluate its functionality and performance.

During testing, users entered questions in natural language through the Streamlit interface. The system successfully interpreted the input and converted it into corresponding SQL queries. These queries were executed on the MySQL database, and the retrieved results were displayed in a structured format on the user interface. The results showed that the system was able to correctly identify relevant database tables and attributes for most user queries.

The experimental results demonstrate that the system simplifies database interaction for users who do not have knowledge of SQL. Users were able to retrieve data using simple English questions without needing to understand database schemas or query syntax. The system also reduced the time required to access information compared to manual SQL query writing.

However, the system may encounter limitations when processing highly complex queries or ambiguous user input. In such cases, additional improvements in prompt design, query validation, and database schema understanding can further enhance the accuracy and reliability of the system.

### Chat Configuration

Host  
localhost

Port  
3306

User  
root

Password  
\*\*\*\*

Database  
employees

LLM Role  
data analyst

Connect

## Smart SQL bot



Hello! I'm a SQL assistant. Ask me anything about your database.



Total no of employees working in this organization?



The total number of employees working in this organization is 300024.

SQL Query: `SELECT COUNT(emp_no) FROM employees`



Give me the list of top 10 employees with the highest current salaries.



The list of top 10 employees with the highest current salaries includes Tokuyasu Pesch with a salary of 158220, Xiahua Whitcomb with a salary of 155709, Tsutomu Alameldin with a salary of 155190, Willard Baca with a salary of 154459, Ibibia Junet with a salary of 150345, Lansing Kambil with a salary of 150052, Juichirou Thambidurai with a salary of 149440, Vivian Minakawa with a salary of 148820, Oldrich Unni with a salary of 147469, and Brigham Teitelbaum with a salary of 146719.

SQL Query: `SELECT e.emp_no, e.first_name, e.last_name, s.salary FROM employees e JOIN salaries s ON e.emp_no = s.emp_no WHERE s.to_date = '9999-01-01' ORDER BY s.salary DESC LIMIT 10`



In which department does this employees with the highest current salary works?



The employee with the highest current salary works in the Sales department.

SQL Query: `SELECT d.dept_name FROM employees e JOIN salaries s ON e.emp_no = s.emp_no JOIN dept_emp de ON e.emp_no = de.emp_no JOIN departments d ON de.dept_no = d.dept_no WHERE s.to_date = '9999-01-01' ORDER BY s.salary DESC LIMIT 1`



What is the average salary in the sales department?



The average salary in the sales department is 89052.0731.

SQL Query: `SELECT AVG(s.salary) FROM employees e JOIN salaries s ON e.emp_no = s.emp_no JOIN dept_emp de ON e.emp_no = de.emp_no JOIN departments d ON de.dept_no = d.dept_no WHERE d.dept_name = 'Sales' AND s.to_date = '9999-01-01'`

Type a message...



FIG.3. Output

Overall, the results indicate that the proposed approach effectively bridges the gap between natural language interaction and structured database querying, making data access more efficient and user-friendly.

## V. CONCLUSION

The proposed system demonstrates an effective approach for enabling users to interact with relational databases using natural language queries instead of traditional SQL commands. By integrating a Streamlit-based user interface with a Large Language Model and a MySQL database, the system allows users to retrieve information from the database in a simple and intuitive way. The language model interprets the user's natural language query and converts it into an appropriate SQL statement, which is then executed on the database to obtain the required results. This approach reduces the complexity of database interaction and eliminates the need for users to have prior knowledge of SQL syntax or database structures.

The implementation of this system shows that natural language processing techniques can significantly improve accessibility to database systems, especially for non-technical users. The system simplifies the process of retrieving information and reduces dependency on database experts. It also helps users obtain the required data more quickly and efficiently compared to manual query writing.

## VI. FUTURE ENHANCEMENT

In the future, the system can be enhanced to support more complex queries involving multiple tables, joins, and nested SQL statements. Additional improvements may include integration with different database management systems, implementation of voice-based query input, and improved query validation mechanisms to ensure higher accuracy and security. The user interface can also be enhanced with advanced data visualization features such as charts and dashboards. These improvements would further expand the usability and efficiency of the system, making database interaction even more accessible and powerful for a wider range of users.

## REFERENCES

- [1] M. Khadija and M. Mustapha, "CHAT-SQL: Natural Language Text to SQL Queries Based on Deep Learning Techniques," *Journal of Theoretical and Applied Information Technology*, vol. 102, no. 12, 2024.
- [2] A. Jha et al., "Conversion of Natural Language Text to SQL Queries Using Generative AI," in *Proc. IEEE Int. Conf. on Computing and Communication*, 2025.
- [3] X. Zhu et al., "Large Language Model Enhanced Text-to-SQL Generation: A Survey," *IEEE Access*, 2024.
- [4] "End-to-End Text-to-SQL Generation within an Analytics Insight Engine," in *Proc. IEEE Conf.*, 2024.
- [5] "LLM-Powered SQL Querying: Transforming Natural Language into Database Insights," *IEEE Journal/Conference Publication*, 2025.
- [6] V. Zhong, C. Xiong, and R. Socher, "Seq2SQL: Generating Structured Queries from Natural Language Using Reinforcement Learning," in *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2017.
- [7] T. Yu et al., "Spider: A Large-Scale Human Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task," in *Proc. EMNLP*, 2018.
- [8] B. Wang et al., "RAT-SQL: Relation-Aware Schema Encoding and Linking for Text-to-SQL Parsers," in *Proc. ACL*, 2020.
- [9] Y. Gan et al., "Natural Language Interfaces to Databases: An Introduction," *ACM SIGMOD Record*, vol. 49, no. 1, pp. 44-51, 2020.
- [10] A. Vaswani et al., "Attention Is All You Need," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
- [11] T. Brown et al., "Language Models Are Few Shot Learners," in *Advances in Neural Information Processing Systems*, 2020.
- [12] R. Popescu, A. Armanasu, O. Etzioni, D. Ko, and A. Yates, "Modern Natural Language Interfaces to Databases: Composing Statistical Parsing with Semantic Tractability," in *Proc. COLING*, 2004.
- [13] P. Liang, M. I. Jordan, and D. Klein, "Learning Dependency-Based Compositional Semantics," *Computational Linguistics*, vol. 39, no. 2, pp. 389-446, 2013.
- [14] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," in *Proc. ICLR*, 2015.

- [15] J. Devlin et al., “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” in Proc. NAACL-HLT, 2019.
- [16] Z. Lin et al., “SQLNet: Generating Structured Queries from Natural Language without Reinforcement Learning,” arXiv preprint arXiv:1711.04436, 2017.
- [17] H. Li and J. Jagadish, “Constructing an Interactive Natural Language Interface for Relational Databases,” Proceedings of the VLDB Endowment, vol. 8, no. 1, pp. 73-84, 2014.
- [18] W. Yao et al., “A Survey on Natural Language Interface to Databases,” Knowledge Engineering Review, vol. 35, 2020.



#### Copyright & License:

© Authors retain the copyright of this article. This work is published under the Creative Commons Attribution 4.0 International License (CC BY 4.0), permitting unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.