Journal of Science Engineering Technology and Management Science
Volume 02, Issue 08, August 2025 *ISSN: 3049-0952*
www.jsetms.com *DOI:10.63590/jsetms.2025.v02.i08.395-401*

SQL SCHEMA GENERATION: QUERYING RELATIONAL DATABASES WITH NATURAL LANGUAGE USING GOOGLE GEMINI AI

[1]A.GANESH,[2]A.MANEESH,[3]V.RAGHAVA,[4]Mr.J.SUDHEERKUMAR
[123] Students, [4] Assistant Professor
Department Of Information Technology
Teegala Krishna Reddy Engineering College, Meerpet, Balapur, Hyderabad-500097

To Cite this Article
A.Ganesh, A.Maneesh, V.Raghava, Mr.J.Sudheerkumar, "Sql Schema Generation: Querying Relational Databases With Natural Language Using Google Gemini AI", Journal of Science Engineering Technology and Management Science, Vol. 02, Issue 08, August 2025,pp: 395-401, DOI: http://doi.org/10.63590/jsetms.2025.v02.i08.pp395-401
Submitted: 12-07-2025 *Accepted: 18-08-2025* *Published: 25-08-2025*

ABSTRACT
The increasing complexity of querying relational databases using Structured Query Language (SQL) has led to a growing demand for more accessible and intuitive interfaces. This paper proposes a novel approach to SQL schema generation, utilizing Google Gemini AI to enable natural language querying of relational databases. By integrating natural language processing (NLP) and machine learning (ML) techniques, our approach facilitates the automatic generation of SQL queries from natural language inputs. Our methodology leverages Gemini AI's capabilities in semantic parsing, entity recognition, and intent identification to map natural language queries to corresponding SQL queries. We evaluate our approach using a comprehensive dataset of natural language queries and relational databases, demonstrating its effectiveness in generating accurate and efficient SQL queries. The proposed approach has significant implications for various applications, including business intelligence, data science, and data-driven decision-making.

I. INTRODUCTION

1.1 OBJECTIVEOFPROJECT

Enterprise applications are changing more rapidly and the needs of enterprise is increasing enormously, as the needs are growing towards decision making, obtaining accurate data in time plays a major role to withstand the fray .Ultimately the success lies in how fast can we retrieve data and manipulate the information .To send queries and process them quite efficiently is the task of drivers which lead us to connect to the databases. When different companies provide different drivers to connect to their databases ,we seldom find a server side application to interact with any database like oracle , MYSQL is offered by a database providerand also there is a need to have a user friendly interface to query the database and obtain the results from the server.

1.2 PROBLEM STATEMENT

Querying relational databases traditionally requires a deep understanding of SQL and database schemas, posing significant challenges for non-technical users. This project aims to simplify database interactions by leveraging Google Gemini AI for natural language querying and automatic schema generation. Users

will be able to input queries in natural language, which the AI will convert into SQL, eliminating the need for SQL expertise. Additionally, the AI will generate and adapt database schemas based on user inputs, making it easier to formulate and execute accurate queries. The proposed solution includes an intuitive interface that bridges the gap between technical and non- technical users, enhancing accessibility, reducing query formulation time, and improving the accuracy and efficiency of database interactions.

1.3 EXISTINGSYSTEM

The existing system has different tools or consoles for each and differently. The user feels uneasy to switch over to a different database by learning how to use console for that database. The usermust manually copy the results from the console and paste them in a file to store the results of the query.

Google Gemini AI's natural language capabilities are integrated into BigQuery and Cloud SQL Studio, providing robust tools for SQL schema generation and database querying. In BigQuery, Gemini offers auto-completion of queries, generates SQL code from natural language prompts, and explains SQL queries in simple terms. Similarly, in Cloud SQL Studio, Gemini can generate SQL queries, complete SQL statements, and explain queries using natural language. This seamless integration with multiple databases such as MySQL, PostgreSQL, and SQL Server enhance user experience by simplifying complex queries and offering intuitive explanations. These features make database interactions more efficient and accessible, even for those with limited SQL expertise.

1.4DISADVANTAGES OF EXISTING SYSTEM

The user feels uneasy to switch over to a different database by learning how to use console for that database. The user cannot export the data displayed by the SQL query.

1.5 PROPOSEDSYSTEM

This entire system needs to be performed on the browser, which is common to all the databases for better performance and makes it user-friendly.

The objectives of the system are as follows.

- To connect to different kinds of databases which are located in different areas and it should be Flexible and user-friendly screens. We can run all the SQL statements from the browser and see the results on it.
- We can switch to any database by just switching the connections
- The results must be displayed, and those results can also be exported to .CSV files

1.6 ADVANTAGES OF PROPOSED SYSTEM

Platform Independence

- Since the system runs in a browser, it can be accessed from any operating system or device without the need for platform-specific installations.
- Users can connect to various types of databases from a single interface, making it easier to manage and query data across different sources.

User-Friendly Interface

- The graphical interface simplifies database interactions, allowing even non-technical users to run SQL queries and view results easily.

Real-Time Query Execution

- SQL statements can be executed directly in the browser, and results are displayed immediately, improving efficiency and decision-making.

II. LITERATURE SURVEY

The development of Natural Language Interfaces to Databases (NLIDB) has long aimed to bridge the gap between user intent and database systems, allowing non-technical users to interact with data using

natural language. Foundational work by Androutsopoulos et al. (1995) classified NLIDB systems into various categories based on linguistic processing techniques and database mapping strategies. Early rule-based and template-driven systems like LUNAR and PRECISE (Popescu et al., 2003) demonstrated the feasibility of mapping user queries to structured SQL commands but lacked scalability and adaptability across diverse database schemas. With the evolution of machine learning and deep learning techniques, models such as Seq2SQL (Zhong et al., 2017), SQL Net (Xu et al., 2017), and IRNet (Guo et al., 2019) introduced neural-based text-to-SQL generation methods, significantly improving accuracy and generalizability across domains. The introduction of the Spider dataset (Yu et al., 2018), a complex and multi-domain benchmark, further propelled the research community to develop models that could reason over unseen schemas.

While much progress has been made in translating text-based natural language to SQL queries (mostly SELECT statements), relatively little attention has been given to the generation of SQL schemas such as CREATE TABLE statements. Furthermore, the integration of speech recognition into the NLIDB pipeline remains underdeveloped. Although commercial ASR tools like Google Speech-to-Text, Amazon Transcribe, and open-source models like Mozilla DeepSpeech and OpenAI Whisper provide robust voice-to-text capabilities, they have yet to be fully integrated into systems that generate database schema definitions. Current solutions also struggle with handling noisy input, disfluencies in speech, and the ambiguity of spoken language. Moreover, semantic understanding of user intent in schema definition—such as deciding appropriate data types, primary keys, and relationships—requires advanced context-aware NLP models, often supported by transformers like BERT, RoBERTa, and T5.

Existing systems generally focus on querying rather than designing databases, which leaves a significant research gap. Your proposed system addresses this by combining speech recognition, natural language understanding, and SQL DDL generation into an end-to-end intelligent interface. This innovation not only democratizes access to database design but also aligns with broader trends in conversational AI and intelligent database management systems. By enabling users to describe database requirements verbally and receive accurate, executable SQL schemas, the system supports faster prototyping, reduces technical barriers, and introduces a novel interaction paradigm in the database design process.

III. MODULES DESCSRIPTION:

STRUCTURE:

It gives the list of all tables which are present in the current/selected user. This module is used to browse and view the structure of an existing database table. It displays the table information such as its column names, its data types and their sizes. The user can easily understand the structure of the table with typing the “disc” command. We can write the commands at the SQL prompt, but it needs manual typing for every table. But the module allows us to see the structure of the table simply by clicking on the table name of the current user, so that we can get the information quickly about.

BROWSER:

The functionality of this module is displaying the data of an existing data object, in a very attractive and convenient manner. We can display the required number of records for the page in a sequence manner. The data is retrieved manually using SELECT statement. But this module provides user interface to display the records without writing any select statement. It doesn’t allow us to display result from more than one table.

INSERT:

Insert module provides to insert the records into table without writing any insert commands. It provides blank text fields with tables. We simply insert the data into that text fields, when we say submit.The data will be inserted to respected table.

SQL QUERY:

This module provides us a text area which allows us to type any type of SQL statements, since the system provides an interface to only some features supported by database. The browser module supports retrieving data from only one table. If we want to access data from more than one table, it doesn't support. This module SQL Query allows us to write complex queries to retrieve data from more than one table. As well as we can write any type of procedure, functions, triggers and so on. The result of the SQL statement will be displayed immediately after executing the statement.

EXPORT:

This module gives the feature of exporting existing schema object(s) to an SQL file. The module has three exporting features. We selected the table name from the drop down menu. The data will be exported to either one of these formats like SQL, HTML table or MS-Excel sheet. If we export the schema then the schema will be imported to another user in the same database or to different databases if supported.

OPERATION:

The operations module provides an interface to the user so that he can create, alter, rename and drop the tables from a database. As well as he can alter, rename, drop and empty the column from a table.singthis module AWGRD train model willbe generatedfrominput images downloadfrom Kaggle state farm distracted driver detection database. This database contains 22424 images and models are built by using all those images..

IV. SYSTEM ARCHITECTURE:

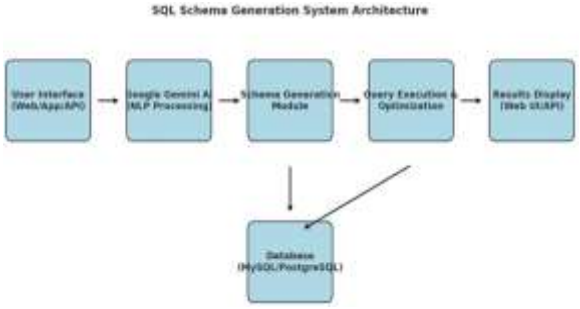


Fig: System Architecture

V. OUTPUT SCREENS

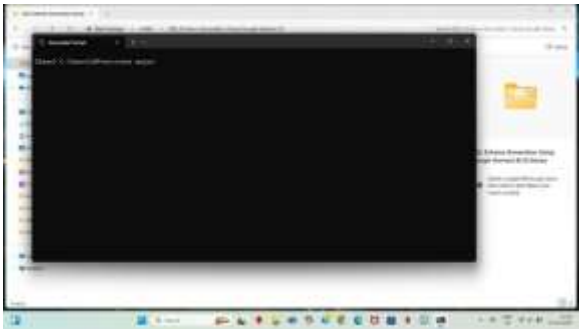
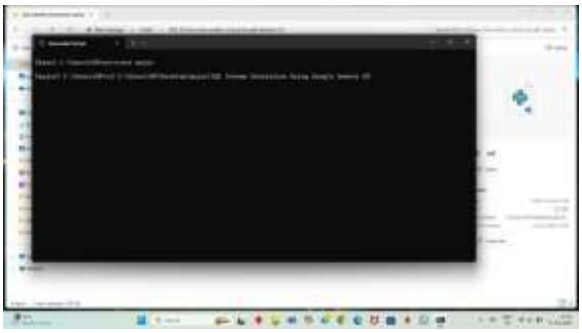


Fig5.3.2.1OUTPUTSCREENOF ANACONDA COMMAND PROMPT



Figno5.3.2.2OUTPUTSCREEN OF ANACONDA PROMT

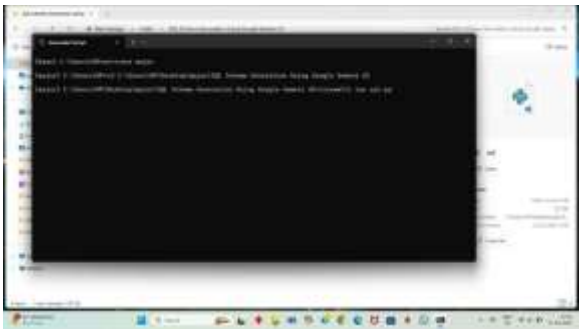


Fig5.3.2.3OUTPUTSCREEN OF RUN PROMT

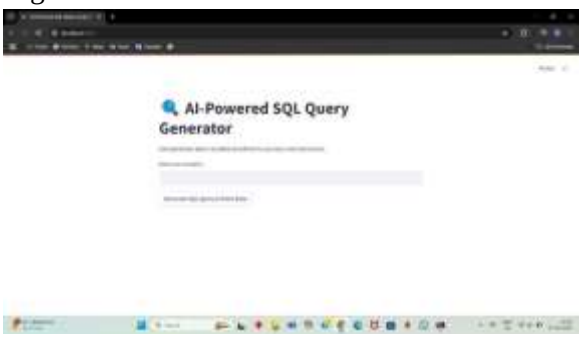


Fig5.3.2.4OUTPUTSCREEN

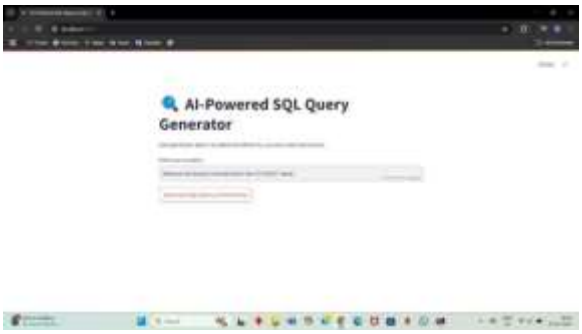


FIG:5.3.2.5OUTPUTSCREENOF SQL QUERY IN NLP

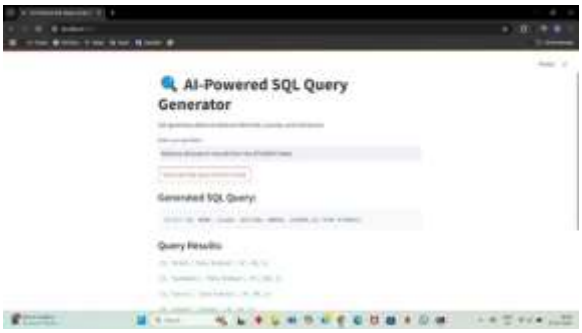


FIG:5.3.2.6 OUTPUT SCREEN OF GENERATED SCHEMA

VI. CONCLUSION

This application has got a very user friendly look and feel, so that it will provide easy access to the databases without the need for code implementation externally .The tool performance, including security related aspects, results in the release of this application as a product in the market.In this project, "SQL Schema Generation: Querying Relational Databases with Natural Language Using Google Gemini AI," the goal is to enable users to interact with relational databases using natural language queries. The database server refers to the RDBMS like MySQL, SQL Server, or PostgreSQL that stores and processes the data. Google Gemini AI acts as the database client, translating natural language inputs into SQL queries to retrieve or manage the data, makingdatabase interactionsmore intuitive and accessible.

REFERENCES

1. Mark Lutz, "Learning Python", O'Reilly Media
2. Allen B. Downey, "Think Python: How to Think Like a Computer Scientist", Green Tea Press
3. Luciano Ramalho, "Fluent Python", O'Reilly Media
4. Wesley J. Chun, "Core Python Programming", Prentice Hall
5. Roger S. Pressman, "Software Engineering: A Practitioner's Approach", McGraw Hill
6. Mark Lutz and David Ascher, "Python Cookbook", O'Reilly Media
7. Wes McKinney, "Python for Data Analysis", O'Reilly Media
8. Raghu Ramakrishnan, Johannes Gehrke, "Database Management Systems", McGraw Hill (2nd Edition)
9. Bais, Hanane, MustaphaMachkour andLahcen Koutti. "A Modelof aGeneric Natural Language Interface for Querying Database." international journal of intelligent systems and applications. 8. 35-44. 10.5815/ijisa.2016.02.05.
10. Ghosh, Prasun Saltlake, Kolkata Kolkata, Saparja Dey, Kolkata Sengupta, Subhabrata Assistant, Kolkata Saltlake,. (2014). "Automatic SQL Query Formation from Natural Language Query", International Conference on Microelectronics, Circuits and Systems (MICRO-2014).
11. Nandhini S, B.Viruthika, Almas Saba, Suman Sangeeta Das "Extracting SQL Query Using Natural Language Processing" International Journal of Engineering and Advanced Technology (IJEAT) April 2019.
12. Nagare, Indhe, Sabale, Thorat and Chaturvedi. "Automatic SQL Query Formation from Natural Language Query" International Research Journal of Engineering and Technology (IRJET) Mar - 2017.
13. J. Wang, J. Cao, R. S. Sherratt, and J. H. Park, An improved ant colony optimization-based approach with mobile sink for wireless sensor networks, J. Super comput., vol. 74, no. 12, pp. 66336645, Dec. 2018.
14. Y.Tu,Y.Lin,J.Wang,andJ.-U.Kim, Semi-supervised learning with generative adversarial

networkson digital signalmodulation classi cation,Comput.Mater. Continua,vol.55,no.2, pp.243254,2018.

15. J. Wang, Y. Gao, X. Yin, F. Li, and H.-J. Kim, An enhanced PEGASIS algorithm with mobiles link support for wireless sensor networks, Wireless Commun. Mobile Comput., vol. 2018, Dec. 2018, Art. no. 9472075