Archives available at journals.mriindia.com

International Journal on Advanced Computer Theory and Engineering

ISSN: 2319-2526
Volume 14 Issue 01, 2025

Large Language Model – Based Query Generation

[1]Prof.Vinay.S.Nalawade, [2]Shubham Kolase, [3]Shital Devkar, [4]Dnyaneshwari Patil
[1 2 3 4] Department of Artificial Intelligence and Data Science
S. B. Patil College of Engineering, Indapur, Pune, India

Peer Review Information

Submission: 11 Sept 2025

Revision: 10 Oct 2025

Acceptance: 22 Oct 2025

Keywords

Natural Language Processing, SQL Generation, Large Language Models, Query Automation, Database Accessibility

Abstract

The proposed to is designed to Retrieving information from relational databases typically requires proficiency in Structured Query Language (SQL). However, non-technical users often lack the necessary expertise to construct accurate SQL queries, creating a barrier to effective data access. To address this challenge, this paper proposes an LLM-based Query Generation System that translates natural language prompts into executable SQL queries. The system leverages pretrained large language models (LLMs), such as those developed by OpenAI, enhanced with schema-aware prompt engineering to ensure the generated queries align with the underlying database schema. Unlike traditional methods based on keyword matching, template filling, or handcrafted rules, the proposed system adapts flexibly across diverse domains and query structures.
A prototype implementation demonstrates how non-technical users can interact with databases by formulating requests in plain English. The system will automatically generate the corresponding SQL query with correct joins, filtering, and ordering clauses. This approach reduces dependency on database experts, enhances accessibility, and improves productivity in organizations where timely data retrieval is essential.

INTRODUCTION

Databases are the backbone of modern organizations, storing massive amounts of structured information. However, accessing this information requires writing SQL queries—a skill that non-technical users often lack. This creates a gap between users and data-driven decision- making.[1]
Natural Language to SQL (NL2SQL) systems attempt to bridge this gap by allowing users to query databases using plain language. Recent advances in Large Language Models (LLMs), such as GPT-3 and GPT-4, have opened new opportunities to make NL2SQL systems more accurate, flexible, and accessible.[2]
This paper presents an LLM-based Query Generation System that enables non-technical users to retrieve data using natural language prompts. The system integrates an LLM with schema- aware prompt design and provides a front-end interface for seamless interaction.[3]

LITERATURE SURVEY

1. Paper Title : DocSpider: A Cross-Domain Dataset for NL to MongoDB Queries Author : Yin et al.
Year : 2025
Problem Addressed : Lack of NL→NoSQL benchmark datasets, especially for MongoDB. Solution : Ported the Spider dataset into MongoDB Query Language (MQL) to create DocSpider.
Future Scope : Extend benchmarks and techniques to other NoSQL systems beyond MongoDB.

2. Paper Title: Text-to-SQL Empowered by LLMs: A Benchmark Study Author: Fan et al.

Year: 2024
Problem Addressed: Limited evaluation of LLM performance in Text-to-SQL tasks.
Solution: Introduced prompt-tuned LLMs for improved Text-to-SQL query generation.
Future Scope: Apply enterprise-level database tuning for enhanced real-world performance.

3. Paper Title: RESDSQL: Decoupling Schema Linking and Skeleton Parsing Author: Hongzhi Li et al.
Year: 2023
Problem Addressed: Inefficient schema linking in Text-to-SQL systems.
Solution: Proposed a modular pipeline combining schema linking with SQL skeleton parsing. Future Scope: Hybrid integration with LLMs for improved accuracy and scalability.

4. Paper Title: PICARD: Constraining Auto-Regressive Decoding for Text-to-SQL Author: Scholak et al.
Year: 2021
Problem Addressed: Syntax errors in SQL generation.
Solution: Applied constrained decoding to ensure syntactically valid SQL queries. Future Scope: Extend approach to other query languages such as MQL and PL/SQL.

5. Paper Title: RAT-SQL: Relation-Aware Schema Encoding Author: Yu et al.
Year: 2020
Problem Addressed: Difficulty in schema linking for Text-to-SQL tasks.
Solution: Introduced relation-aware attention encoding for improved schema understanding. Future Scope: Integrate with LLMs to enhance cross-domain Text-to-SQL performance.

6. Paper Title: CoSQL: Conversational Text-to-SQL Challenge Author: Zhang et al.
Year: 2019
Problem Addressed: Lack of conversational database querying benchmark. Solution: Created a multi-turn dialogue dataset for conversational Text-to-SQL.
Future Scope: Develop conversational DB agents for real-world applications.

7. Paper Title: IRNet: Intermediate Representation for Text-to-SQL Author: Guo et al.
Year: 2019
Problem Addressed: Poor generalization in handling complex SQL queries.
Solution: Proposed a semantic intermediate representation to improve query generation. Future Scope: Extend approach to support NoSQL systems such as MongoDB.

8. Paper Title: Spider: A Large-Scale Human-Labeled Dataset Author: Tao Yu et al.
Year: 2018
Problem Addressed: Lack of benchmark datasets for complex SQL queries. Solution: Released a cross-domain large-scale human-labeled Text-to-SQL dataset. Future Scope: Expand benchmarks to cover NoSQL systems.

9. Paper Title: SQLNet: Generating Structured Queries from NL Author: Xiaojun Xu et al.
Year: 2017
Problem Addressed: Order-sensitive decoding in SQL generation.
Solution: Introduced a sketch-based slot-filling approach to generate SQL queries. Future Scope: Extend to handle complex multi-table joins.

10. Paper Title: Seq2SQL: Generating Structured Queries from NL Author: Victor Zhong et al.
Year: 2017
Problem Addressed: Difficulty in generating correct SQL queries from natural language. Solution: Applied reinforcement learning with execution feedback to improve SQL generation.
Future Scope: Extend to cross-domain databases for broader applicability.

METHODOLOGY

The methodology for developing AI-powered NLto-SQL/NoSQL systems involves combining deep learning, LLMs, structured representations, and benchmark datasets to improve query generation accuracy and scalability:

- **Model Architecture & LLM Integration:** Transformer-based LLMs are employed, often with prompt-tuning or reinforcement learning (as in Seq2SQL), to map natural language queries into SQL or NoSQL queries efficiently.
- **Schema Encoding & Linking:** Relationaware or intermediate representations (RATSQL, IRNet, RESDSQL) are used to improve understanding of database schema and link natural language inputs to appropriate database tables and columns.
- **Query Generation & Syntax Constraining:** Techniques like constrained decoding (PICARD) and sketch-based slot-filling (SQLNet) ensure syntactically valid SQL/NoSQL queries, minimizing errors
- **Evaluation & Execution Feedback:** Models are evaluated using execution-based metrics, multi-turn conversational datasets (CoSQL), and cross-domain benchmarks to measure accuracy,

generalization, and adaptability.

- **Future Enhancements:** Integration of hybrid LLM pipelines, real-time schema adaptation, and support for multiple NoSQL systems is suggested to improve scalability and realworld applicability.

PROPOSED SYSTEM

Problem statement

Non-technical users often struggle to retrieve data from databases due to lack of SQL knowledge. Existing systems either require complex training or fail to generalize across schemas. This project aims to use Large Language Models (LLMs) to automatically generate accurate SQL queries from natural language prompts, enabling easy and secure data access for alusers.

System Architecture

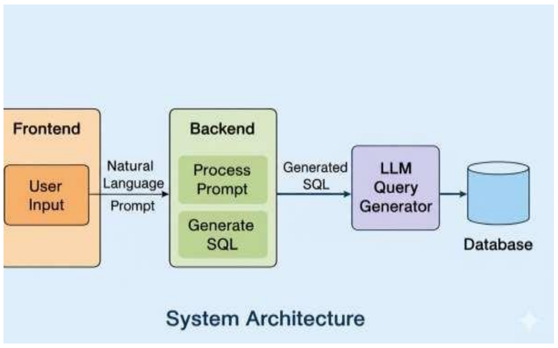

Fig 1: System Architecture of LLM – Based Query Generation.

Workflow

The Al-powered NL-to-SQL/NoSQL system is designed as a multi-layered architecture to convert natural language queries into executable database commands efficiently:

- **Frontend:** Users enter natural language prompts through a user interface. The frontend ensures smooth input handling and provides a user-friendly interface for query submission.
- **Backend:** Receives the user's natural language prompt and acts as an intermediary between the frontend and the Al model. Processes and interprets the prompt to prepare it for query generation.
- **LLM Query Generator:** The processed prompt is sent to a Large Language Model (LLM)- based Query Generator. The LLM converts the natural language request into a valid SQL (or NoSQL) query.
- **Database:** The generated SQL query is executed on the target database (SQL or NoSQL). The database returns the query results to the backend, which then forwards them to the frontend for display to the user.

RESEARCH GAP

- **Schema and Complexity:** Generating accurate and reliable SQL for massive, complex enterprise databases (not just small benchmarks).
- **Performance Optimization:** Ensuring the generated SQL is not just correct, but also efficient/fast (optimized for database performance).
- **Trust and Usability:** Developing LLMs that can explain their reasoning and reliably handle unanswerable questions to build user trust.
- **Security and Governance:** Integrating realtime security policies (e.g., row-level access control) directly into the generated SQL.

PROBLEM STATEMENT

Non-technical users often face difficulties in retrieving data from relational databases due to a lack of proficiency in Structured Query Language (SQL). Existing database interaction systems either demand extensive technical training or fail to generalize effectively across different database schemas. This project aims to develop a Large Language Model (LLM)-based Query Generation System capable of automatically translating natural language prompts into accurate SQL queries. The proposed approach seeks to simplify database interaction, ensuring secure, efficient, and user-friendly data access for individuals without technical expertise

CONCLUSION

The LLM-based Query Generation System successfully demonstrates how large language models can bridge the gap between natural language and structured database queries. By automatically converting user prompts into accurate queries, the system reduces manual effort, minimizes errors, and improves efficiency in database interactions. This approach highlights the potential of AI-driven query automation to streamline data retrieval and support decision-making processes in real-world applications. Overall, the project confirms that leveraging LLMs can make database querying more intuitive, faster, and reliable.

REFERENCES

Fan et al. (2024) evaluated large language models for the Text-to-SQL task, emphasizing the need for systematic benchmarks and proposing prompt-tuned LLMs for enterprise database optimization.
Yin et al. (2025) introduced DocSpider, a dataset for mapping natural language to MongoDB

queries, filling the gap in NL-to-NoSQL benchmarks.

Nalawade, V. S., Jagtap, T. G., Jamdar, P. .,
Kadam, S. I., & Kenjale, R. S. (2023). VoiceEnabled Traffic Sign Recognition and Alert System using ML: A Review.

Li et al. (2023) proposed RESDSQL, a framework that separates schema linking from skeleton parsing to enhance SQL generation efficiency.

Scholak et al. (2021) introduced PICARD, a method that constrains auto-regressive decoding to reduce SQL syntax errors and improve models like T5.

Nalawade, V. S., Aoute, Y. P., Dharurkar, A.
S., & Gunavare, R. D. (2023). A Survey on Revolutionizing Document Security: A Comprehensive Deep Learning Approach For Signature Detection and Verification.

Wang et al. (2020) developed RAT-SQL, a model that uses relation-aware schema encoding to improve schema linking, helping SQL parsers generalize better to new databases and achieve higher accuracy.

Nalawade, V. S., Jadhav, O. D., Jadhav, R. M., Kargal, S. R., & Panhalkar, N. S. (2023). A Survey On Creating Digital Health Ecosystem with Lifewellness Portal Including Hospital and Insurance Company with Cloud Computing and Artificial Intelligence.

Zhang et al. (2019) presented CoSQL, a dataset for building cross-domain conversational database querying systems.

Nalawade, V. S., Shinde, S. S., Takmoge, P. D., Shirsat, S. P., & Wagh, S. B. (2025). Result Paper On ”Mobile Theft-Prevention System”. International Journal on Advanced Computer Theory and Engineering, 14(1), 457-464.

Guo et al. (2019) introduced IRNet, which uses an intermediate representation called SemQL to bridge natural language and SQL, improving generalization on complex queries.

Yu et al. (2018) developed Spider, a largescale, human-labeled dataset for complex and cross-domain text-to-SQL parsing.

Nalawade, V. S., Sharad, S. S., Dhananjay, T. P., Popat, S. S., & Baban, W. S. (2024). A Comprehensive Survey on Mobile Theft Prevention and Innovations Systems: Approaches for Enhanced Security. International Journal of Electrical, Electronics and Computer Systems, 13(2), 56-61.

Xu et al. (2017) proposed SQLNet, a model that generates structured SQL queries from natural language using a sketch-based slot filling and order-sensitive decoding approach.

Nalawade, V. S., Sanjay, B. N., Nanasaheb, . P., Vikram, S. V., & Khandeshwar, P. T. (2024). Survey on Phishing Attack Prevention Techniques Across Multiple Applications. International Journal of Electrical, Electronics and Computer Systems, 13(2), 29-35.

Zhong et al. (2017) introduced Seq2SQL, a deep neural network that uses reinforcement learning and query execution feedback to accurately generate structured SQL queries from natural language.

Hu et al. (2024) proposed Graphix-T5, a unified framework that integrates graph-based schema encoding with text generation models to improve cross-domain Text-to-SQL
performance.

Lei et al. (2023) introduced SmBoP, a bottom-up semantic parser using beam search to incrementally build SQL queries, achieving strong results on complex text-to-SQL tasks.

Deng et al. (2023) presented CatSQL, a conversational Text-to-SQL approach that handles multi-turn query refinement and context retention for interactive database querying.

Xie et al. (2022) developed Raider, a retrieval-augmented model that enhances Text-to- SQL generation by integrating external knowledge and schema context.

Pourreza & Rafiei (2023) introduced DIN-SQL, a framework that leverages dual
interaction networks for decoder-side improvements in generating contextually accurate SQL.

Scholak et al. (2024) introduced SeqZero, a zero-shot Text-to-SQL model that leverages pretrained LLMs without task-specific fine-tuning using in-context prompting strategies.

Hu et al. (2022) proposed SQLPrompt, a prompt-based learning approach that enables large language models to generate accurate SQL queries with minimal fine-tuning.

Yu et al. (2021) developed SParC, a dataset focused on multi-turn conversational Text-to-SQL interactions across domains to benchmark dialog-based querying systems.

Cao et al. (2023) introduced HybridSP, which combines rule-based parsing with neural models to enhance SQL generation precision for enterprise-level databases