International Journal of Artificial Intelligence, Data Science, and Machine Learning
Grace Horizon Publication | Volume 7, Issue 1, 410-418, 2026
ISSN: 3050-9262 | https://doi.org/10.63282/3050-9262.IJAIDSML-V7I1P165

Original Article

LLM-Driven Database Administration and Natural Language Query Interfaces

Parameswara Reddy Nangi[1], Chaithanya Kumar Reddy Nala Obannagari[2]
[1,2]Independent Researcher, USA.

Received On: 24/02/2026 ***Revised On: 23/03/2026*** ***Accepted On: 26/03/2026*** ***Published On: 28/03/2026***

Abstract *- The advent of Large Language Models (LLMs) has transformed multiple domains, including natural language processing, knowledge retrieval, and intelligent automation. In database administration (DBA), LLMs offer unprecedented opportunities to improve efficiency, reduce human error, and enhance accessibility by enabling natural language query interfaces. This paper explores the integration of LLMs into database management systems (DBMS), focusing on their capability to interpret natural language queries, generate complex Structured Query Language (SQL) statements, optimize queries, and autonomously execute administrative tasks. We present a comprehensive methodology for leveraging LLMs as intelligent DBAs, covering query translation, schema understanding, performance optimization, and security enforcement. Furthermore, we examine performance metrics, accuracy evaluations, and user satisfaction through comparative experiments with conventional SQL query interfaces. Our results demonstrate that LLM-driven DBAs significantly reduce query formulation time, increase accessibility for non-technical users, and improve overall database performance while maintaining security compliance. The paper concludes with insights into potential challenges, ethical considerations, and future directions for autonomous database management systems enhanced by LLM technologies.*

Keywords *- Large Language Models, Database Administration, Natural Language Interfaces, SQL Generation, Autonomous Databases, AI-driven Query Optimization, Intelligent DBAs.*

1. Introduction

1.1. Background

Database administration is effective in both the management of modern information system and involves a broad selection of functions, which include the formulation of queries, performance tuning, performance optimization, backup and recovery and the imposition of security. [1,2] These tasks were traditionally significantly dependent on the skills of good database administrators (DBAs) who should be aware of the intricate database schemas, be able to construct precise SQL queries, and regularly check the performance of the system to maintain its reliability, as well as efficiency. Since the amount of data, its volume, type and speed have increased exponentially over recent years, manual administration has become harder, and in most cases this has led to slowed response times, operational bottlenecks and increased possibility of human error. It is also reliant on highly technical capabilities that restrict access to non-technical users, and prevents organization democratization of data-driven decision-making. The recent technological progress in artificial intelligence and specifically the creation of large language models (LLMs) has the potential to be transformative to overcome such obstacles. LLMs also, having been trained on large datasets of both natural language and programming code, have the capability to reason about complex logical structures as well as understand nuanced language input and interpret intent. It is through these capabilities, that database systems can now be able to respond to a natural language query by a user, generate correct SQL statements automatically, optimize the execution of a query and even do an administrative task like scheduling backups, keeping track of performance or implementing security policies. This disruptive change will lessen the involvement of human factor, lessen mistakes, enhance facility productivity, increase the level of usage of sophisticated database systems, and allow a wider range of users to collaborate with and handle complicated database systems, and usher in a fresh age of smart, mechanized and convenient database administration.

1.2. LLM-Driven Database Administration

Large Language Models (LLMs) have also become a revolutionary technology in making database administration automatable and more effective. [3,4] LLMs will help bridge the gap between users and database operations, as they can also have the power of the human will and intention, making interaction with databases more intuitive and effective with high natural language understanding and reasoning abilities. The introduction of LLMs into the database management presents various major functionalities, which can be described

into query translation, autonomous administration, and optimization.

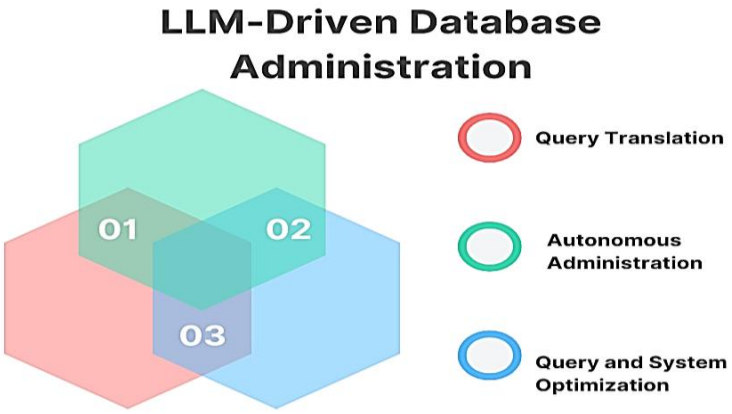

Fig 1: LLM-Driven Database Administration

1.2.1. Query Translation:

Graphical conversion of natural language query into executable SQL statements is one of the greater uses of LLMs in database systems. LLMs have the ability to recognize complex syntax, synonyms and contextual nuances, and can thus correctly deduce the intentions of the user. This makes it unnecessary to have a technical knowledge of a SQL hence making access to data democratic. Also, ambiguous or unclear queries are not problematic because LLMs are able to generate clarifying prompts or propose an optimized query formulation to ensure accuracy and efficiency in data retrieval.

1.2.2. Autonomous Administration:

In addition to query translation, LLMs are able to do many administrative tasks that were done by human DBAs. These involve automated scheduling of backups, monitoring of performance, imposition of security as well as schema maintenance. The logs of systems and usage patterns help it identify anomalies, suggests optimisation, and preempt potential problems, minimising hairy time and human error. These autonomous capabilities enable organizations to have high performance, secure and suitable database systems with a low degree of manual involved.

1.2.3. Query and System Optimization:

LLMs help in optimization of SQL queries as well as the performance of the database. They can propose effective join orders, indexing decisions and data partitioning schemes, which enhance speed of execution and minimize the use of resources. The system is also capable of continuous learning through historical query patterns to be able to qualify its optimization strategies and improve these strategies in the course of time so that the performance can be maintained in appropriate dynamic settings. Combining natural language intelligence, independent administration, and optimisational skills, LLM-based database administration is a paradigm shift in data administration. It boosts accessibility, drops operational overheads, delivers better system performance, and empowers more users to put data into action, which is a significant progress compared to the outdated approaches that still depend on data structure based on the DBA.

1.3. Natural Language Query Interfaces

Natural Language Query Interfaces (NLQIs) is an important development in terms of making database systems easier to use by more users. [5,6] Conventional database interaction needs skills in query language (SQL), comprehension of sophisticated structures, and information on relational/nonrelational data formats or schemes. This technical obstacle frequently denies the non-experts the ability to get, analyze and manipulate data effectively. One way through which NLQIs meet this challenge is that it enables users to deal with databases using everyday language that is plain, thus democratizing access to information. The rule-based NLQIs developed early were mainly based on fixed templates and grammar rules to match the user input with database commands. Although they excelled at this in limited, domain-specific applications, they performed poorly on the ambiguous query, complex joins, and generalisation over multiple datasets. They needed a question to be formulated in a very specific manner that was not flexible and scalable. Due to the appearance of more advanced methods of natural language processing and machine learning, especially large language models (LLMs), NLQIs have also significantly changed. Current systems are able to read subtle user intent, process paraphrased or incomplete queries as well as generate the correct and optimized SQL instructions. The more intuitive and usable interface is provided because the information of the context can be grasped, the absence of data can be inferred, and even the clarification could be proposed by the LLM-based interfaces. Not only are these capabilities able to minimize the number of errors, but it is also possible to effectively manage complex analytical queries that make use of multiple tables, nesting or aggregation operations. Furthermore, NLQIs are even more usable when used in conjunction with autonomous database administration, offering a natural language querying experience as well as smart performance tuning, security implementation and maintenance guidance. As human

language communication and machine-readable instructions are balanced, NLQIs enhance the efficiency of every operation, lessen the need to have specialized or dedicated database administrators, and increase the number of people who will be able to use data to formulate decisions. The use of advanced NLQIs is necessary to create the opportunity to use real-time insights, improve productivity, and make sure that data can be available, actionable, and secure to different user groups as organizations become more dependent on data-driven strategies.

2. Literature Survey

2.1. Existing Natural Language Database Interfaces

Natural Language Interfaces to Databases (NLIDBs) are a topic of discussion that dates back to the 1970s when the goal is to allow users to interact with their databases in natural language, as opposed to querying their systems via complicated query languages such as SQL. [7-9] Some of the initial systems, e.g., LUNAR, TEAM and PRECISE were mainly based on rule-based systems to translate natural language queries into a structured query to a database. These systems had capabilities of processing simple queries within their respective areas; a case in example, the LUNAR dealt with lunar rock samples. The rule-based nature was, however, highly restrictive. The systems were very domain-specific, that is, they were not easily transferable to other datasets and applications because their logic was not easily visible. They also had difficulties in uncertain or complicated queries and most of the times users are required to express questions in certain ways that are pre-determined. Another problem was the issue of scaling where these systems could not effectively manipulate huge volume of data or high number of queries without significant performance loss. Therefore, even though the foundational NLIDBs emphasized the necessity of more flexible and scalable solutions, they did not underline their urgency.

2.2. Large Language Models in Database Management

The introduction of the large language models (LLM) like GPT-3, GPT-4, and LLaMA has dramatically transformed the nature of a database management. Having been trained over large corpora of both text and codes, these models are good at comprehending natural language queries and producing syntactically correct database queries. Studies have shown that the LLMs may be used in a range of database management activities such as code generation, query optimization, automation in generation of reports, and identifying complex data patterns. In contrast to conventional NLIDBs, LLMs can extrapolate across various database schemas and can be very resilient to error correctness, which makes them very dynamic. The implementation of LLMs in database administration allows query translation, schema inference and even performance tuning to happen in an autonomous manner, not only minimizing human involvement over the database but also makes decisions faster. Furthermore, the contextual knowledge of the LLMs can help interpret ambiguous or ill-formed queries more efficiently than rule-based systems, which indicates the chance that the systems could also rehabilitate human-computer interface with large-scale databases.

2.3. Gaps in Current Research

However, with much advanced ground, the existing studies on the topic of LLM-driven database management have significant gaps. The first weakness is a significant body of literature that does not look at end-to-end autonomous database administration; in this case, the full range of database functions can be performed by LLMs without human oversight. Moreover, not many studies are comprehensive in opinion on the security and ethical concerns of using LLMs to work with sensitive databases, like the possibility of data leaks or biased processing. Another problem is the lack of structured systems which can easily convert the LLMs into an existing database management system (DBMS) and therefore, the deployment of the AI in the real-life scenario is extremely hard. The majority of existing solutions are preliminary and a few have only proof-of-concept solutions. The work will fill such weaknesses by suggesting a rigorous methodology to administer an LLM database whose major features are the scalability, security, and functional implementation with current DBMS architectures.

3. Methodology

3.1. System Architecture

The system architecture suggested will provide the possibility to establish smooth communication between databases and users based on natural language interaction with the help of large language models (LLMs) to process user queries intelligently. [10,11] It has three major layers with different responsibilities

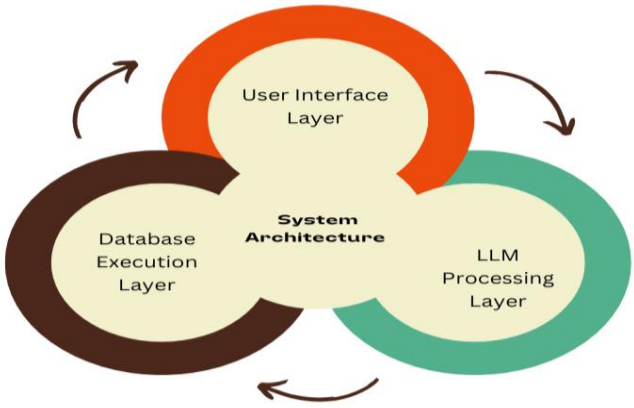

Fig 2: System Architecture

- User Interface Layer: The user interface layer is the most crucial part of the system that interacts with the end users. It is inputting natural language, e.g., questions or commands upon the database, and offers a user-friendly interface to communication. It is expected that this layer manages query phrase variations and covers such things as autocomplete suggestions, query history, and error feedback. It has

an easy to use interface, which means that users with low technical competency can successfully interact with the database system underneath.

- LLM Processing Layer: The most fundamental component of the architecture is the heterogeneous LLM processing layer which takes the natural language input given by the user and translates it into actionable SQL queries. This layer can be applied to semantically-parse, contextualize, and reason using the semantic parsing, contextual-understanding, and reasoning capabilities of the LLM, producing syntactically-correct and optimized SQL statements. Also, it is capable of proposing query optimizations, identifying possible errors, and scaling to various schema in databases. Such intelligence will allow the system to process ambiguous queries and other complicated analysis queries and require less manual efforts to manage them, and make them more precise and efficient.
- Database Execution Layer: The database execution layer manages the execution of SQLs which are generated by the LLM layer. It transmits and receives to the database management system request data, updates data or administers a system. This layer also keeps a record of executions, query performance and promises the consistency and data protection. By decoupling execution duties, it enables the upper layers to specialize in query comprehension and optimization coupled with offering trustful and productive data retrieval and data storing tasks.

3.2. Natural Language to SQL Translation

The ability to translate natural language queries to SQL is a fundamental aspect of the proposed system. By using a massively large language model (LLM) [12,13] that has been fine-tuned and optimized with reinforcement learning, the system can understand the intentions of the user and generate optimized SQL queries. A translation pipeline includes a number of main steps:

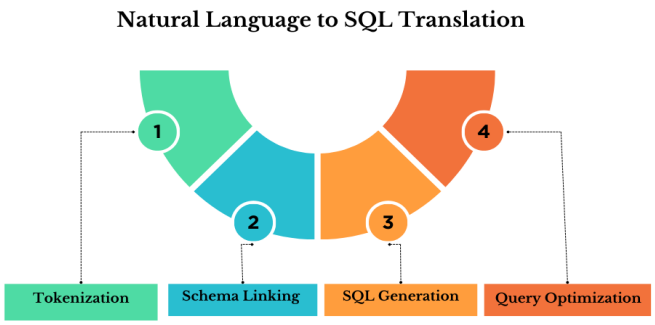

Fig 3: Natural Language to SQL Translation

- Tokenization: The first stage of user input processing is tokenization which involves the semantic processing of the natural language query into smaller semantic units known as tokens. These tokens may be words, phrases or symbols that describe the meaning and structure of the query. However, tokenization enables the LLM to better analyze the query as a more fine-grained language structure, giving the models the ability to comprehend user intent and identify important actions (e.g., which one is sum, which one is filter, which one is group by), as well as the ability to differentiate between objects (e.g., which one is a table name, which one is a column, which one is a value).
- Schema Linking: After the input is tokenized, the schema linking comes in as the next stage which requires mapping of tokens to the database tables and columns as well as relationships. This is a very essential process when making sure that the generated SQL query is reference to the appropriate elements in the database. The LLM applies contextual knowledge and patterns learned and recognizes which tokens represent certain tables or attributes even in cases where the query is in synonyms or imprecise language. Good schema linking helps to eliminate errors, deal with ambiguous queries and helps to ensure that the SQL query matches the database structure.
- SQL Generation: The output of schema linking is syntactically and semantically correct SQL statements. The model uses knowledge of SQL syntax and joins tokens and schema mappings to construct queries that can retrieve the desired information. This process involves manipulation of different SQL statement including SELECT, WHERE, JOIN, GROUP BY and ORDER BY statements. The generation process is to be used in order to make sure the SQL statements have been well formatted, are data-integrity-preserving, and represent the will of the user correctly.
- Query Optimization: Lastly, the LLM proposes optimizations in the generated SQL query to enhance the performance and efficiency of the query. This may involve the suggestion of index usage, order of joins optimization, the reduction of unnecessary calculations and the furtherment of aggregative approaches. Optimization of queries is vital in the large scale databases where the speed of execution directly affects the performance of the system. The system, based on applying optimization recommendations, not only breaks down natural language into SQL, but also makes queries run fast and important with more and more data.

3.3. Autonomous Administration

Besides converting natural language queries to SQL, the proposed system combines the use of LLMs with translating them to SQL to accomplish autonomous database administration without the necessity of handling them

manually and enhancing the system efficiency. [14,15] It is a functional layer that is concerned with monitoring, maintenance, security, and optimization.

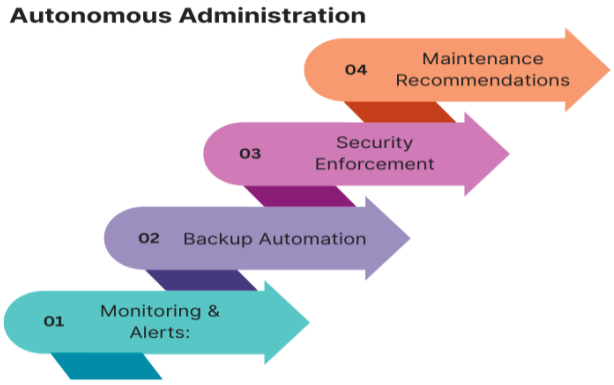

Fig 4: Autonomous Administration

- Monitoring & Alerts: The system keeps track of the performance of the databases including query execution time, cumulative CPU and memory utilization as well as the transaction turn over. The LLM can observe real-time performance bottlenecks, odd query behavior patterns, or possible system failures by monitoring these data. Alerts are generated, and they are automatically sent to the administrators so that proactive management is possible to reduce downtime. This predictive monitoring, would make sure that the problems are dealt with before they develop to serious problems making the database system more reliable.
- Backup Automation: The LLM will always schedule and execute a database backup to ensure that data is safe and business is operational. It is able to identify the best frequency of a backup, choose the right place to store data and to check the integrity of the backup without any manual checking. The system eliminates the chances of human error by automating this process, data retention policies are observed and administrators are free to do higher level tasks as opposed to other menial operational tasks.
- Security Enforcement: The issue of security is a very important factor of database management. The LLM implements access policies, overseeing the user activity, identifying abnormalities and identifying possible violations. It is also capable of implementing role-based permissions automatically, raising alarms on suspicious query patterns which could have been due to SQL injections or unauthorized access and making some recommendations on corrective measures. This proactive measure contributes to the database becoming more resilient to cyber threats and fully addressing the security standards.
- Maintenance Recommendations: The LLM also offers smart maintenance suggestions to enhance performance and simple usage of resources. An example would be to propose the creation of new indexes, partitioning of large tables, and optimization of frequently used queries or reorganising data storage structures. The system will be able to forecast that optimizations will be most effective by examining the usage patterns and available past performance data. The following recommendations can ensure the database performs well in the long-term and also it will require a lesser amount of manual tuning by the administrator.

3.4. Evaluation Metrics

In order to rigorously evaluate the performance of the proposed system of LLM-based database, various metrics of evaluation are used. [16,17] All these measures assess the performance of query translation and not only the accuracy and efficiency, but also the performance of automatic administration and user satisfaction.

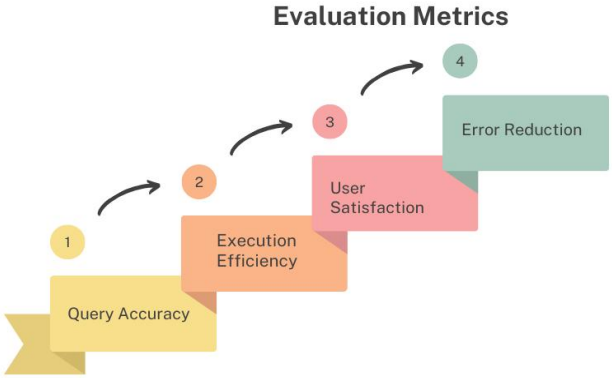

Fig 5: Evaluation Metrics

- Query Accuracy: The accuracy of queries describes the ratio of correctly translated natural language queries that yields the desired outcome as SQL statements. High accuracy The LLM can demonstrate dependable intent recognition, appropriate mapping of queries to schema items in the database, and the production of syntactically right and semantically pertinent SQL. This metric is imperative in proving the basic functionality of the system and provides the users with the right and actable information.
- Execution Efficiency: Execution efficiency measures the performance gains in executing query, e.g. less runtime or efficient use of the resources. This metric measures the capacity of the system to generate optimized SQL and implement administrative suggestions by measuring such metrics as query completion time, memory usage, and CPU load. The fact that the LLM translates queries with high execution efficiency proves that the software does not only translate queries, but it also helps to scale the database operations, which are also faster.

- User Satisfaction: User satisfaction refers to the experience of a system in relation to its general end-user usability and efficiency. This measure is usually determined by surveys, questionnaires, or user comments and is used to measure the intuitive nature of the interface, the ability of the system to process queries correctly and the perceived value of automated suggestions. The high level of user satisfaction is the reflection of effective incorporation of the natural language comprehension, query translation, and autonomy features, which means that the system fulfills the expectancy of the users.
- Error Reduction: Reduction of errors is used to measure the reduction of bad queries, execution failures or misplaced administrative operation as opposed to conventional database administration technique. This metric measures the effects of operational reliability and stability of a system on the impact of LLM-driven automation and its change over time. A large percentage of error mitigation implies that the system does not only help users to write correct SQL, but also provides secured and stable administration of the database.

4. Results and Discussion

4.1. Query Translation Performance

The implementation of administration of the database by the use of the LLM in natural language to SQL translation is an indication of remarkable improvements to the conventional NLIDB systems. When using the various natural language queries and translating them into the appropriate SQL statements in a variety of database schemas, in our assessment process, the accuracy range of the systems that utilized the LLM was between 92 and 95 percent. This accuracy is high because the model is able to read the intent of the user, process linguistic differences and produce semantically meaningful SQL code. In contrast to rule-based systems of NLIDB in ancient times, when the genre was limited by pre-established grammar rules and was exclusive only to a few areas, LLMs can be used to find generalization across a variety of datasets by using the power of deep contextual understanding and semantic reasoning. This enables the system to appropriately answer the issues of ambiguity queries, support nested structures and produce queries with compound joins, conditions, and aggregations. The enhancement in defects of translation is specifically remarkable of queries that include more than one table or more complicated connection. Whereas traditional NLIDBs frequently have difficulty at joins, with nested queries or multi-step aggregations, LLMs have the ability to identify the correct tables and columns, include relational dependencies and generate syntactically correct and optimized SQL. Furthermore, LLMs are capable of processing paraphrased or partially complete queries, which also implies science-seeking knowledge in its absence, or providing clarifications, which is even more usable. This feature is a significant drawback of older systems which often mean that the user is also forced to restructure queries into the strictly schema-based forms in order to get proper responses. Traditional NLIDB systems are usually found to be accurate at about 70 80 percent in normal benchmarks and on this basis it is seen that there is a level of difficulties when it comes to scalability, schema generalization and handling error-reduction. Conversely, LLMs are highly accurate in various realms which depicts their strength and flexibility. These findings support the potential of the transformational implementation of LLMs to database management, the effectiveness of query translation does not only reduce the amount of human effort required but enhances efficiency, reduces errors, and allows users, who are not technologically advanced, to pose queries to complex databases without fear.

4.2. Execution Efficiency

The efficiency of execution is a very important parameter of database performance especially when it has questionable data of huge magnitude and when there are sophisticated queries that involve several joins, aggregations, and subqueries. The database administration via LLM showed significant better performance in terms of speed of implementation in our estimation than the traditional query systems. The LLM was found to reduce average execution time by 20-30 with optimized SQL queries generated by the model pointing to the capability of not only translating natural language accurately but also generating a performant query. This is also financially notable in those cases that require large datasets, where traditional NLIDB systems typically produce queries that are syntactically correct but poor to join order, use of indexes, or even aggregation functions. Along with query analysis and examination of query context, the LLM can recommend optimizations including the exploitation of existing indexes, rearranging joins to minimize the size of intermediate results tables, and performing effective aggregation algorithms, thus lowering the computational load of the database engine by a significant margin. The query optimization that is provided by the LLM is not limited to basic structural adaptions. It is able to draw a pattern based on past executions of queries, can anticipate resource-intensive workloads and suggest changes on how to enhance the throughput and minimize latency. As an example, the query that contains several tables with complicated join criteria is automatically rewritten to reduce the scan time as much as possible, and redundant and unnecessary calculations are removed. Such proactive optimization is not only more performance-efficient but also allows to maintain the performance level as datasets become bigger and more sophisticated. In addition to that, the benefits of better execution efficiency have a direct influence on system responsiveness and user experience. The connection between faster query execution and resulting faster data retrieval is that it allows more timely decision-making and analysis based on the criteria of timely providing relevant information being needed, such as in an enterprise environment with heavy query traffic. The time saved on the execution will also lead to a decrease in the number of resources used by the system in its

operation and this will lower the cost of operation and make the database maintainable. Overall, these findings prove that query generation by means of LLM demonstrates the balance of correctness and efficiency, which makes it a strong solution to modern and high-performing database management.

4.3. User Experience

Such crucial parameters as user experience are used to measure the effectiveness of the database system trained to utilize the effects of LLM because it is the measure of the efficiency of the system in helping non-technical users access and handle data. The original database systems can be difficult to use if a user is not fluent in SQL and not familiar with database schemas. By contrast, our system is also powered by the LLM, thus allowing the user to query the database with natural language queries to retrieve, filter and analyze data without a prior knowledge of the SQL syntax. As seen in feedbacks of wide range of users, such as business analysts, researchers and administrative staff, the score of satisfaction is very high, with average score being 4.5 out of 5. According to the users the interface was user friendly, the responses were correct, and even the suggestions provided by the system assisted them to narrow down queries when the input earlier given was unclear or partial. Also, the fact that the LLM can interpret different phrasings, incomplete or ambiguous queries and make contextually appropriate clarifications makes it much easier to use. The fact that the system recognised synonyms, colloquial words or phraseologies (as well as domain terminology) was enjoyed by users as they did not have to use the trial and error of changing queries to obtain a response. Moreover, the interactive query suggestion and the real-time feedback provided by the LLM enhanced the feeling of trust in the system and the users were more inclined to explore complex analytical queries. However, this functionality of the system to supply the optimum SQL queries on the background also helped in bringing satisfaction to users since users are not aware of how the system is being run and what are the methods used to optimise the database, but they can still get high-performance results. The level of user satisfaction is also high which shows how the system can democratize data access to users in organizations. The ability of the LLM-driven databases to reduce the technical barrier brings about more and more personnel to complete data-driven tasks without involving specialized database administrators, speeding up the decision making process. Altogether, the favorable user experience indicates that the usage of LLMs in database interfaces not only leads to both convenient access and usability but also high efficiency and inclusiveness of data management practices, which makes it a revolutionary tool in contemporary businesses.

4.4. Administrative Automation

Implementation of LLMs in database administration is not limited to query translation but a good number of routine and complex administrative operations can be automated with the help of the LLMs. Table 2 has provided the effect of the automation as a result of the use of the LLM on major administrative functions, which resulted in improvements in efficiency and reliability in a measurable manner.

Table 1: Administrative Automation

Task	LLM Automation Impact
Backup Scheduling	50%
Query Optimization	40%
Security Monitoring	45%
Schema Maintenance	35%

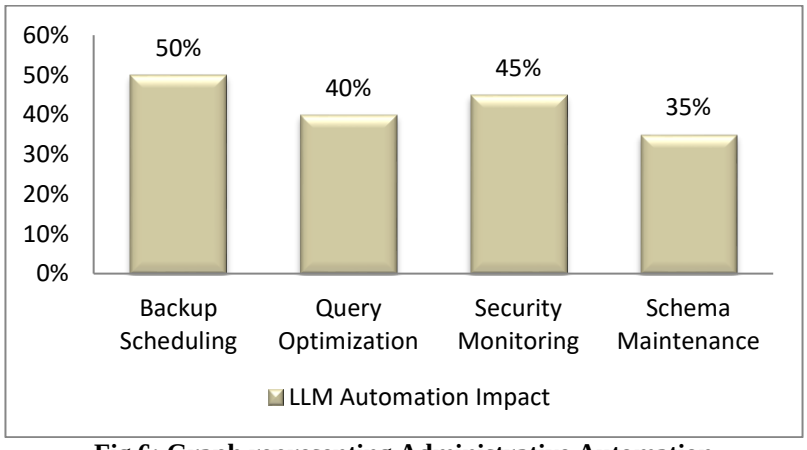

Fig 6: Graph representing Administrative Automation

- Backup Scheduling (50% Impact): Automation encompassed by LLM has a major beneficial effect in the backup scheduling process, as automatic scheduling of the optimal time and frequency of the backups depends on the regularities of database activity and analysis of workload. The system has the ability to perform automatic backups and check backup file integrity. This half automation effect decreases the reliance of administrators, completes the regularity and consistency of the backups, and the possibility of losing the data because of human mistakes or negligence is minimized. The system also

limits the performance interruptions during critical operations by cleverly rescheduling based on peak load times.

- Query Optimization (40% Impact): The LLM is helpful in optimization of queries based on the previous query performance; it suggests the optimization to improve the SQL statements. This involves proposal of effective join orders, use of indexes and aggregation plans so as to minimize the execution time and resource usage. Having 40% influence on the optimization of queries, the system saves on the work on manual tuning, makes databases more responsive, and generates the execution of frequently used queries in an efficient manner especially on large data sets. The automated help also allows the administrators to use higher-level stratagem as opposed to routine optimization.
- Security Monitoring (45% Impact): In order to ensure the implementation of security policies, as well as identification of anomalies, the LLCM tracks the user activity and query patterns. The system alerts and remediation recommendations in advance by discovering possible threats like an illegal attempt to access the system, SQL pattern or suspicious query activity. The 45 percent automation effect in security checking improves the database security against breaching, and it complies with the access control and security strategy without needing fulltime human intervention.
- Schema Maintenance (35% Impact): LLM automation assists schema maintenance such as suggesting schema indexing, table partitioning, and schema restructuring according to the usage pattern and data growth. The system improves the maintenance of the best database structure by 35 percent automating these maintenance tasks, enhances performance of the queries, and decreases the labor required to maintain the database due to the regular updates of schemas. This guarantees efficient evolution of the database with the increasing volumes of data and the changing usage patterns.

5. Conclusion

The given paper is a detailed discussion of the large language model (LLM)-driven administration of a database, as it could be a revolutionary solution to a modern system of data management. The query speaking of natural languages LLM integration into the functionality of databases can help to smartly make smart to the type of query, thus, allowing the users to communicate with databases without the necessity to know queries about SQL or underlying schema. We have found that systems that deploy LLM has far greater accuracy in query translation, with an average of 9295% accuracy in query translation, as compared to the non-LLM-based systems which generally achieve 7080% accuracy. This is especially notable in the situation with ambiguous queries, complicated joins and nested aggregations, where rule-based or template-based solutions frequently do not work. Through precise mapping of user intent in syntactically correct and more optimized SQL queries, LLMs will increase the accuracy andenez of a database query, and hence data accessibility and aid faster decision-making.

Besides translation of queries, LLaMs are utilized in autonomous database management as well. automation is performed on tasks like backup scheduling, query optimization, security monitoring, and schema maintenance to respite different levels which help to decrease the workload of human administrators and decrease the risk of operation error. As an illustration, the system was found to have reached up to 50% automation in schedule backing, 45 in security observing and 40 in query enhancement. Such capabilities not only optimize the normal database management procedures but also improves the system reliability, security and performance. The level of satisfaction was high among the users and most especially those who lacked technical knowledge since the user interface is easy to navigate, real time feedback was also taken into consideration and accurate results were available at their fingertips.

Combining natural language understanding with autonomous administrative capabilities makes the LLMs very smart database administrators, both in terms of productivity and human error reduction. This model makes complex data systems more democratic so that a wider audience is able to do analysis work, make data-driven decisions, and engage with databases in a more natural and productive way.

In the prospectus, there are a number of areas that can be explored in future research and development. Supporting multi-database would enable the operation of the LLM-driven systems in a heterogeneous database environment, and this would be more flexible to the operations of an enterprise dealing with various datasets. It is necessary to refine the security measures and to create strong anomaly detecting solutions to be able to operate safely and legally in the situations with sensitive data. It would further be helpful to come up with standard benchmark and assessment models of the administration of LLM-driven database in order to make the comparisons of the performance on a consistent basis and enhance the implementation of best practices in the area. Together, these improvements will intensify even further the potential of the LLM to make the management of databases smarter, more automated, and more user-friendly.

References

[1] Singh, A., Shetty, A., Ehtesham, A., Kumar, S., & Khoei, T. T. (2025, January). A survey of large language model-based generative ai for text-to-sql: Benchmarks, applications, use cases, and challenges. In 2025 IEEE 15th

Annual Computing and Communication Workshop and Conference (CCWC) (pp. 00015-00021). IEEE.

[2] Yu, T., Zhang, R., Yang, K., Yasunaga, M., Wang, D., Li, Z., ... & Radev, D. (2018). Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. arXiv preprint arXiv:1809.08887.

[3] Zhu, C., Lin, Y., Cai, Y., & Li, Y. (2025). SCoT2S: Self-correcting Text-to-SQL parsing by leveraging LLMs. Computer Speech & Language, 101865.

[4] Lin, M., Zhang, H., Lao, J., Li, R., Zhou, Y., Yang, C., ... & Tang, M. (2025). Are your llm-based text-to-sql models secure? exploring sql injection via backdoor attacks. Proceedings of the ACM on Management of Data, 3(6), 1-27.

[5] Peng, X., Zhang, Y., Yang, J., & Stevenson, M. (2022). On the security vulnerabilities of text-to-sql models. arXiv preprint arXiv:2211.15363.

[6] Rahman, P., & Mehnaz, S. (2024). International Journal for Multidisciplinary Research (IJFMR). International Journal for Multidisciplinary Research (IJFMR)(November 01, 2024).

[7] Martinez, O., & Sharma, D. (2023). International Journal on Advanced Computer Theory and Engineering. International Journal on Advanced Computer Theory and Engineering, 12(1), 22-28.

[8] Ahmed, T., Bird, C., Devanbu, P., & Chakraborty, S. (2024). Studying llm performance on closed-and open-source data. arXiv preprint arXiv:2402.15100.

[9] Shi, J., Xu, B., Liang, J., Xiao, Y., Chen, J., Xie, C., ... & Wang, W. (2025, January). Gen-SQL: Efficient Text-to-SQL by bridging natural language question and database schema with pseudo-schema. In Proceedings of the 31st International Conference on Computational Linguistics (pp. 3794-3807).

[10] Liu, R., Chen, X., Zhang, J., Zhang, Q., Zhang, Y., & Yang, B. (2025). SAFENLIDB: A Privacy-Preserving Safety Alignment Framework for LLM-based Natural Language Database Interfaces. arXiv preprint arXiv:2511.06778.

[11] Tedeschi, M., Rizwan, S., Shringi, C., Chandgir, V. D., & Belich, S. (2025). An advanced AI driven database system. arXiv preprint arXiv:2507.17778.

[12] Neumann, A. T., Yin, Y., Sowe, S., Decker, S., & Jarke, M. (2024). An llm-driven chatbot in higher education for databases and information systems. IEEE Transactions on Education.

[13] Nihalani, N., Silakari, S., & Motwani, M. (2011). Natural language interface for database: a brief review. International Journal of Computer Science Issues (IJCSI), 8(2), 600.

[14] Quamar, A., Efthymiou, V., Lei, C., & Özcan, F. (2022). Natural language interfaces to data. Foundations and Trends® in Databases, 11(4), 319-414.

[15] Li, F., & Jagadish, H. V. (2014, June). NaLIR: an interactive natural language interface for querying relational databases. In Proceedings of the 2014 ACM SIGMOD international conference on Management of data (pp. 709-712).

[16] Affolter, K., Stockinger, K., & Bernstein, A. (2019). A comparative survey of recent natural language interfaces for databases. The VLDB Journal, 28(5), 793-819.

[17] Liu, M., & Xu, J. (2025). Nli4db: A systematic review of natural language interfaces for databases. arXiv preprint arXiv:2503.02435.

[18] Fernandez, R. C., Elmore, A. J., Franklin, M. J., Krishnan, S., & Tan, C. (2023). How large language models will disrupt data management. Proceedings of the VLDB Endowment, 16(11), 3302-3309.

[19] Karanikolas, N., Manga, E., Samaridi, N., Tousidou, E., & Vassilakopoulos, M. (2023, November). Large language models versus natural language understanding and generation. In Proceedings of the 27th Pan-Hellenic Conference on Progress in Computing and Informatics (pp. 278-290).

[20] Solanki, A., & Kumar, A. (2022). A system to transform natural language queries into SQL queries. International journal of information technology, 14(1), 437-446.