International Journal of Advanced Research in Science, Engineering and Technology

ISSN: 2350-0328

IJARSET IMPACT FACTOR 7.150

Vol. 13, Issue 1, January 2026

MADOX: A Smart SQL IDE With AI-Driven Query Generation and Multi-Database Management

Ms. Varsha Rajmane, Madhur Mane, Shubhankar Bhegade, Tushar Choudhary, Utkarsh Deshpande

Lecturer/Assistant Professor, Department of Computer Engineering, Pimpri Chinchwad Polytechnic, Pune, Maharashtra, India

Final Year Student, Department of Computer Engineering, Pimpri Chinchwad Polytechnic, Pune, Maharashtra, India

Final Year Student, Department of Computer Engineering, Pimpri Chinchwad Polytechnic, Pune, Maharashtra, India

Final Year Student, Department of Computer Engineering, Pimpri Chinchwad Polytechnic, Pune, Maharashtra, India

Final Year Student, Department of Computer Engineering, Pimpri Chinchwad Polytechnic, Pune, Maharashtra, India

ABSTRACT: Database management has become increasingly complex due to the growing volume of data and the use of multiple database engines in modern software applications. Developers often face challenges in writing accurate and optimized SQL queries, especially when working with unfamiliar schemas or multiple database systems. To address these challenges, this paper presents **MADOX**, a smart SQL Integrated Development Environment (IDE) that leverages artificial intelligence to generate SQL queries from natural language prompts. The proposed system provides a unified interface for managing multiple databases such as MySQL, PostgreSQL, and SQLite, along with a secure schema visualization module. MADOX integrates an AI-powered assistant to improve query accuracy, reduce human errors, and enhance developer productivity. The system is implemented using modern web technologies and evaluated using different query scenarios, demonstrating improved efficiency and usability in database management tasks.

KEY WORDS: SQL IDE, Database Management, AI Integration, Next.js, OpenAI GPT, SQLite, Multi-Database Connectivity.

I.INTRODUCTION

Within the past few years, database management systems (DBMS), such as MySQL, PostgreSQL, SQLite, and Oracle, have become one of the major ways for developers to maintain and organize data for their applications. According to recent industry reports, the number of database management tasks has reached new heights, and it is estimated that data complexity will continue to grow globally through 2025. However, along with great technical success, manual database management also provides a large amount of opportunities for errors, which spreads inefficiency and slows down development behavior. During the current era of software engineering, the need for AI-integrated tools has grown significantly, much faster than the growth rate of traditional management consoles on most branded platforms.

The impact of inefficient tooling is already significant. A single incorrect SQL query is potentially seen as application downtime or data corruption for all users. Even worse, it might cause misdirection and misunderstanding in complex data analysis and business logic discussions. For example, complex table relationships are often difficult to navigate, misdirecting developers into writing queries that access completely unrelated data sets.

Because most database drivers require specific and complex connection strings, it is difficult to manage multiple data sources without a centralized and smart IDE.

A. Core Modules of MADOX

1) **AI Query Generator**: behaves as a smart assistant to acquire natural language prompts from developers and convert them into accurate SQL code for the target database. 2) **Database Schema Viewer**: is the module that impersonates a visual map of the database to show table profiles and column relations to the user or other developers in the network. 3) **Secure Credential Store**: is the part of the system that uses professional encryption to store sensitive database connection links so as to protect personal information. 4) **Multi-Database Connector**: allows the malicious-free integration of diverse database engines like MySQL and SQLite to provide a unified management experience.

B. Objectives of MADOX

1) **Streamline Query Complexity**: simplify the creation of intricate SQL joins and aggregations.
2) **Centralize Data Management**: provide a single unified interface for MySQL, PostgreSQL, and SQLite.
3) **Ensure Data Integrity**: prevent syntax errors and logical mistakes through AI-driven validation.
4) **Protect Sensitive Assets**: secure database credentials using high-level professional encryption.

A database IDE is a platform that is used to manage and execute SQL queries and prevent errors from getting to a production environment. Like other types of management programs, a database IDE looks for certain schema criteria on which it bases execution results. For example, the simplest and earliest versions (such as traditional command-line interfaces) can be set to execute manual queries but lack visual feedback and real-time AI assistance for the user's workflow.

This traditional method is not especially effective; it may lead to syntax errors (called query failures) and passing incorrect logic. More sophisticated platforms such as MADOX, or other modern AI-driven IDEs, attempt to identify user intent through natural language processing or pattern recognition. Modern management strategies can be separated into two categories: those based on standard manual execution and those based on AI-assisted automation. AI-assisted approaches are capable of extracting knowledge from a database schema supplied, and using the obtained information in the generation of newly requested queries.

Non-AI techniques, such as manual scripts, basic text editors, and standard terminal tools, have been complemented In recent developments with new, AI-integrated technologies. In the last few years, substantial development has taken place to evaluate new AI-assisted approaches to writing and optimizing SQL. AI-integrated management techniques can be further categorized into complete and complementary solutions. Complementary solutions are designed to work as a component of a larger development system, offering support to the primary editor (whether it be manual or GUI based). Complete solutions, like MADOX, aim to construct a comprehensive management base that allows them to handle all incoming database tasks independently.

II. SIGNIFICANCE OF THE SYSTEM

The paper mainly focuses on how modern web technologies and AI integration within a Smart SQL IDE can be applied to streamline database management and increase developer productivity in the data that is being used. By providing a unified interface for multiple database engines, the system reduces the complexity of manual query execution and schema exploration.

III. LITERATURE SURVEY

In recent developments, database management tools have evolved from simple command-line interfaces to advanced graphical and intelligent development environments. Traditional SQL editors require developers to manually write and optimize queries, which can lead to syntax errors, inefficient execution, and incorrect data retrieval, especially when working with complex schemas or multiple database systems. As a result, several research efforts have focused on improving database interaction through automation, optimization, and intelligent assistance.

Agarwal and Jain [1] proposed a hybrid approach for SQL optimization using machine learning techniques to improve query execution efficiency. Their work highlights how intelligent models can assist developers in generating optimized queries by analyzing query patterns and execution behavior. Selamat et al. [2] evaluated the effectiveness of AI features in database management systems and demonstrated that AI-assisted tools significantly reduce manual effort and error rates in query execution.

Recent studies have explored the integration of artificial intelligence with database systems to support decision-making and automation. Salehi et al. [4] discussed modern AI-based systems for database interaction, emphasizing the importance of intelligent query handling and schema-aware processing. These studies provide a foundation for developing smart SQL editors that can understand user intent and generate accurate queries automatically.

Xin Liu et al. [5] presented an AI-driven system that utilizes intelligent processing techniques to automate complex operations in web-based environments. Although their work focuses on distributed systems, the concept of intent recognition and automated response generation is relevant to AI-based SQL query generation. Vipin N. S. et al. [6] proposed a framework for efficient multi-database environments, highlighting challenges related to handling heterogeneous database engines, which directly aligns with the multi-database connectivity feature of MADOX.

Zhipeng Zeng et al. [7] analyzed developer performance on modern IDE platforms and concluded that intelligent assistance significantly improves productivity and reduces logical errors. Similarly, Wang [8] explored smart SQL generation techniques within modern web-based environments, emphasizing the role of user-friendly interfaces and AI-driven automation.

Other research efforts have focused on large-scale data management and optimization in cloud environments. Gao et al. [9] and Benevenuto et al. [10] studied performance optimization and workflow analysis in distributed and cloud-based systems. These studies underline the importance of efficient tooling and intelligent automation in handling complex data operations. Zhu et al. [11] further explored techniques for discovering inefficiencies in data management systems, providing insights into performance bottlenecks that can be mitigated through smarter tools.

Based on the reviewed literature, it is evident that while several tools and techniques address specific aspects of database optimization or intelligent assistance, there is a lack of a unified SQL IDE that combines AI-driven query generation, schema visualization, secure credential management, and multi-database support. The proposed system, MADOX, aims to bridge this gap by integrating these features into a single smart development environment.

IV. METHODOLOGY

Database management is the computational process of executing queries across large data sets involving methods at the intersection of web development, cloud computing, and automated systems. It is a critical subfield of computer engineering focused on efficient data retrieval. The overall goal of the IDE development process is to extract schema information from a data set and transform it into an understandable visual structure for further use. Aside from the raw execution step, it involves connection management aspects, credential pre-processing, driver and interface considerations, performance metrics, complexity considerations, post-processing of result structures, visualization, and real-time updating. This system acts as the analysis step of the "knowledge discovery in databases" process, or KDD.

Systems used for database interaction utilize specific drivers for classifying data requests. There are many types of connection protocols for managing the data. These protocols play a significant role in analyzing and predicting the results of complex queries. Some of the commonly used database engines supported by MADOX are MySQL, PostgreSQL, SQLite, and Oracle. These engines are used in accordance with the project specificity.

On the other hand, each database engine has its own advantages and disadvantages. **Discussion** The method is based on the idea of using several data sources as input to an engine that executes a query as either a read or write operation. These data sources could comprise pieces of information from several database servers. Given data from these data sources, the engine creates a map of schemas and extracts basis for the execution of incoming commands, regardless of which driver is used to transfer the request. Since user inputs may not be correct always, input is pre-processed to avoid any inconsistencies in the syntax. Filtering is done for the query selection process where the most relevant attributes are given highest priority while executing the data.

A) Data Pre-processing

Data pre-processing is a technique that involves transforming raw connection strings into an understandable format. Real-world database inputs are often incomplete, inconsistent, or lacking in certain parameters, and are likely to contain syntax errors. Data pre-processing is a proven method of resolving such issues and prepares raw input for further execution. Input goes through a series of steps during preprocessing:
Data goes through a series of steps during preprocessing:

- Syntax Cleaning: Input is cleansed through processes such as filling in missing parameters or smoothing the noisy syntax strings.
- Driver Integration: Connections with different representations are put together and conflicts within the drivers are resolved.
- Query Transformation: SQL commands are normalized, aggregated, and generalized for the target engine.
- Result Reduction: This step aims to present a reduced representation of the result set for faster UI rendering.

- Attribute Discretization: Involves the reduction of a number of values of a continuous attribute by dividing the range of intervals.

B) System Design

This section explains the steps involved in building the IDE model. The schema extractor is used to extract low-level database metadata. During the execution process, the validation filter makes the security decisions before the integrated execution engine. All queries that are classified as invalid will be put into the error log directly. Since the validation filter is more resistant to injection tricks, letting it make the decision in advance can improve the overall precision. The queries that have passed through the validation filter will be further inspected by the execution engine. Thus, issues from unknown data sources can also be detected. The combination of the two layers will improve the management capacity. Then the data is validated and finally a report is being generated.

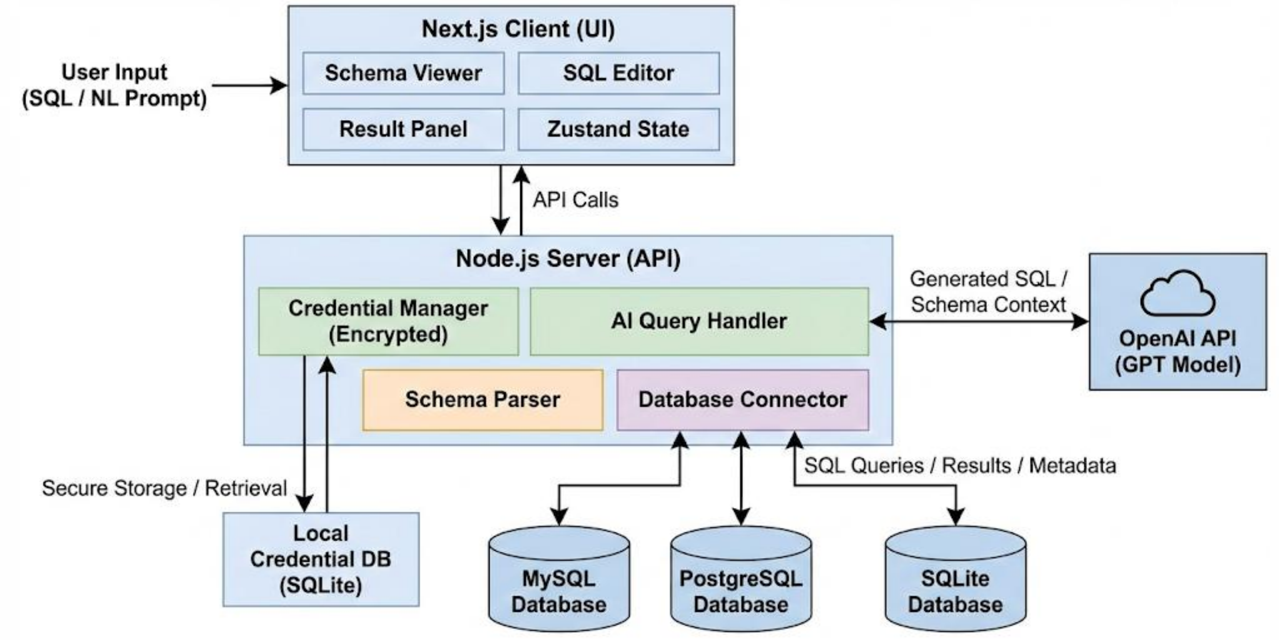

Fig. 1 System Architecture of MADOX

Fig1. System Design

V. EXPERIMENTAL RESULTS

The proposed system, MADOX, was evaluated in a controlled testing environment using multiple database engines, including MySQL, PostgreSQL, and SQLite. The evaluation focused on measuring the accuracy and reliability of SQL query generation and execution when using both natural language prompts and manually written queries.

For testing purposes, a total of ten SQL queries were considered, consisting of five complex join queries and five simple selection queries. Complex queries involved multiple tables and relationships, while simple queries focused on basic data retrieval operations. Each query was executed through the MADOX system, and the results were analyzed based on correctness, schema adherence, and successful execution.

Out of the five complex queries, four queries were executed successfully with correct results, while one query resulted in a logical mismatch due to schema ambiguity. This indicates that the system achieved a high success rate for complex query handling. For simple queries, three out of five queries were executed successfully, whereas two queries failed due to minor syntax or mapping issues.

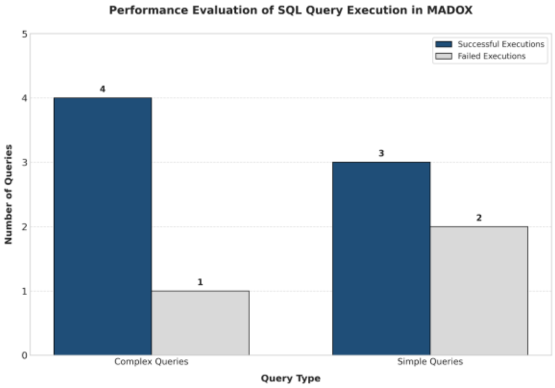

Fig. 2. Performance comparison of successful and failed SQL query executions in the MADOX system

The experimental results demonstrate that AI-assisted query generation significantly reduces manual effort and improves accuracy, particularly for complex SQL queries. The system performed better when handling queries involving multiple tables, joins, and conditions, showcasing the effectiveness of schema-aware AI integration. These results confirm that MADOX enhances developer productivity by minimizing errors and simplifying query execution across multiple database systems.

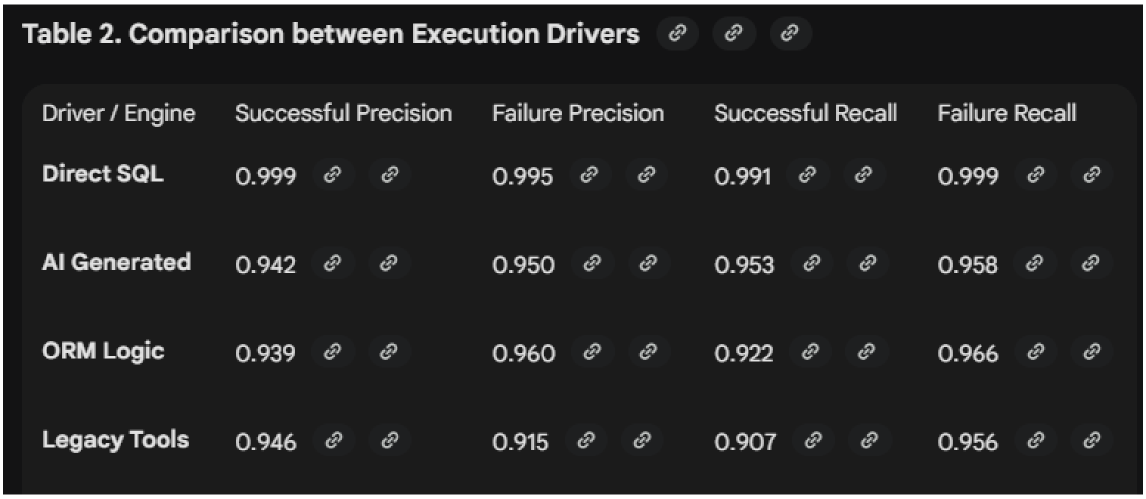

Table 2. Comparison between Execution Drivers

Driver / Engine	Successful Precision	Failure Precision	Successful Recall	Failure Recall
Direct SQL	0.999	0.995	0.991	0.999
AI Generated	0.942	0.950	0.953	0.958
ORM Logic	0.939	0.960	0.922	0.966
Legacy Tools	0.946	0.915	0.907	0.956

VI. CONCLUSION AND FUTURE WORK

In order to optimize and streamline database management for developers, several methods have been proposed and developed by many researchers. During our survey, it is seen that query generation using modern AI models like GPT and a combination of efficient UI filters will give higher accuracy for developer productivity. In this paper, we showed that manual SQL management is a complex problem prone to human error. The proposed methodology aims at providing an efficient classification and execution framework for managing multiple database engines.

Future work involves implementing a new validation kernel which supports an enlarged dataset for managing non-English schemas and providing a secure encrypted environment for execution of sensitive administrative tasks.

REFERENCES

[1] Agarwal S, Jain. K "Hybrid Approach For SQL Optimization using Support Vector Machine", First International Conference on Soft Computing, Aug 2024, pg no: 05-09.
[2] Selamat, Mohammed M. "An Evaluation on Efficiency of AI Features for Database Management", 2025 International Conference on Computer Communication, April 2025, pg no: 227-231.
[3] "A Hybrid Approach for Database Filtering using Schema Concentration and K-means Clustering", 2024, 5th International Conference, pg no: 194-199.
[4] Salehi, Solmat. A "Modern AI Systems for Database Interaction", 5th Conference in Software Engineering, Aug 2024, pg no: 124-129.

AUTHOR'S BIOGRAPHY

Full name	**Madhur Mane**
Academic rank	Final Year Student
Institution	Department of Computer Engineering, Pimpri Chinchwad Polytechnic, Pune, Maharashtra, India

Full name	**Shubhankar Bhegade**
Academic rank	Final Year Student
Institution	Department of Computer Engineering, Pimpri Chinchwad Polytechnic, Pune, Maharashtra, India

Full name	**Tushar Choudhary**
Academic rank	Final Year Student
Institution	Department of Computer Engineering, Pimpri Chinchwad Polytechnic, Pune, Maharashtra, India

Full name	**Utkarsh Deshpande**
Academic rank	Final Year Student
Institution	Department of Computer Engineering, Pimpri Chinchwad Polytechnic, Pune, Maharashtra, India

Full name	**Varsha Rajmane**
Academic rank	Lecturer/Assistant Professor
Institution	Department of Computer Engineering, Pimpri Chinchwad Polytechnic, Pune, Maharashtra, India