Comparing LLM-Generated SQL Query Performance

Kayla Afonesca, Arafat Ahmed, Avinash Bissoondial, Olivia Cava, Ryan Saklad,
and Irene Taylor
Worcester Polytechnic Institute
CS 542: Database Management Systems
Professor Mohammed Al-Kateb
Aug 9, 2024

Table of Contents

Introduction

Database Management Systems (DBMS) have been the backbone of information storage and retrieval for decades. It allows for handling massive amounts of data efficiently. The standard language used by DBMS to query and retrieve information is Structured Query Language (SQL). Traditionally, SQL queries have been written by humans to organize, retrieve and update data, regardless of the data's size or complexity.

However, with the recent rapid advancements in Artificial Intelligence (AI) across various fields, a significant development has emerged in the realm of database management systems: Large Language Models (LLMs). These AI-driven models, originally created for tasks like translation, are now being used to generate SQL queries. This creates a new way to interact with data. Rather than queries needing to be crafted by hand, these AI-powered systems offer to transform natural language queries into SQL. This allows employees without programming experience the ability to analyze data deeply.

Our project aims to evaluate the effectiveness of LLMs in generating SQL queries and to compare their performance with queries written by experienced database professionals. We will also explore whether these LLMs have the potential to help organizations save time and resources in real-world applications, ultimately assessing their value in modern data management.

Related Works

Large Language Models are increasingly being used to generate, fine-tune, and rewrite SQL queries, and this concept has attracted a significant amount of research interest. In a 2024 study published by Li et al., the authors propose an LLM-enhanced query rewire system that optimizes the selection of rewrite rules based on pre-existing databases (Li et al., 2024). The proposed system would improve the overall efficiency of SQL queries through the use of LLMs, where they would suggest strategies for effective revisions to be applied within a standard database management framework. The authors of this study also made sure to address drawbacks, including the common tendency for LLMs to hallucinate, causing them to generate incorrect queries. All in all, experimental results showed that their method significantly improved the query efficiency and it outperformed the baseline methods they measured against (Li et al., 2024). This research is incredibly relevant to what our group aimed to research, because we wanted to assess the performance of various different LLMs in creating optimal SQL queries. While the study by Li et al. focused more on query rewriting strategies through the use of LLMs, our team aims to research and learn about which LLMs are most successful in generating the most optimal SQL queries.

Another fascinating study by Schullhoff et al. in 2024 delves into the wide world of Generative Artificial Intelligence, and really focuses into the many facets of prompting techniques (Schulhoff et al., 2024). More specifically, the authors of this study focused on prompting techniques across various modalities. Their work involves understanding prompts first through a taxonomy with which the goal was to standardize confusing terminology that is currently prevalent in the field of prompt engineering. This taxonomy includes 33 vocabulary terms, 58 text-based prompting techniques, and 40 techniques spanning other modalities. The overarching message of this research was that there is a need for a robust framework that can adapt to the complex natures of LLM applications (Schulhoff et al., 2024). This study informed our process, as it helped us to realize the importance of structuring our queries in a robust, comprehensive manner.

One interesting application of integrating LLMs into SQL queries in a business context is highlighted in a 2024 study done by Painter et al.. In this study, the authors discuss the integration of OpenAI's GPT-4 into pharmacovigilance (PV) to better enhance drug safety (Painter et al., 2024). According to the World Health Organization, pharmacovigilance "is the science and activities relating to the detection, assessment, understanding and prevention of adverse effects or any other medicine/vaccine related problem" (Pal, 2024). This study demonstrated that using LLMs within this particular business context document significantly improves the accuracy of SQL queries (Painter et al., 2024). Initial tests did unfortunately show a low success rate with the database schema, but after integrating more contextual information, the outcomes improved by a landslide. These findings showcase the potential to make SQL querying more intuitive and efficient, and removes barriers for entry for those who

do not already have the strong skills for query writing, optimization, and more (Painter et al., 2024). This is relevant to our project work, because the authors of this study utilized GPT-4 as their LLM. Our group aims to examine and compare differences between various LLMs, not just GPT-4, in their abilities to generate and optimize SQL queries.

Lastly, a study by Liu and Mozafari in 2024 had a significant influence on the trajectory of our project, as it too involved query rewriting through the use of LLMs (Liu and Mozafari, 2024). The authors of this paper present a system for leveraging LLMs to improve efficiency of SQL query rewrites called "GenRewrite." One way they gave hints to guide the LLM was through Natural Language Rewrite Rules (NLR2s), where the LLM had a chance to learn from each interaction. Their system was interesting and unique because it uses counterexamples as a guide for correction and it iteratively refines the query rewrites. Some of their empirical results showed that GenRewrite surpassed some existing query rewriting tools, because it was able to deliver faster and more accurate query rewrites (Liu and Mozafari, 2024). This research, similar to the research conducted by Li et al., is very relevant to our project in the sense that it involved utilizing LLMs to optimize SQL query rewrites. Our study diverges from these pre-existing ideas, however, as it not only uses LLMs for optimizing SQL queries, but it also investigates which LLMs generate the best, most optimal queries.

Methods

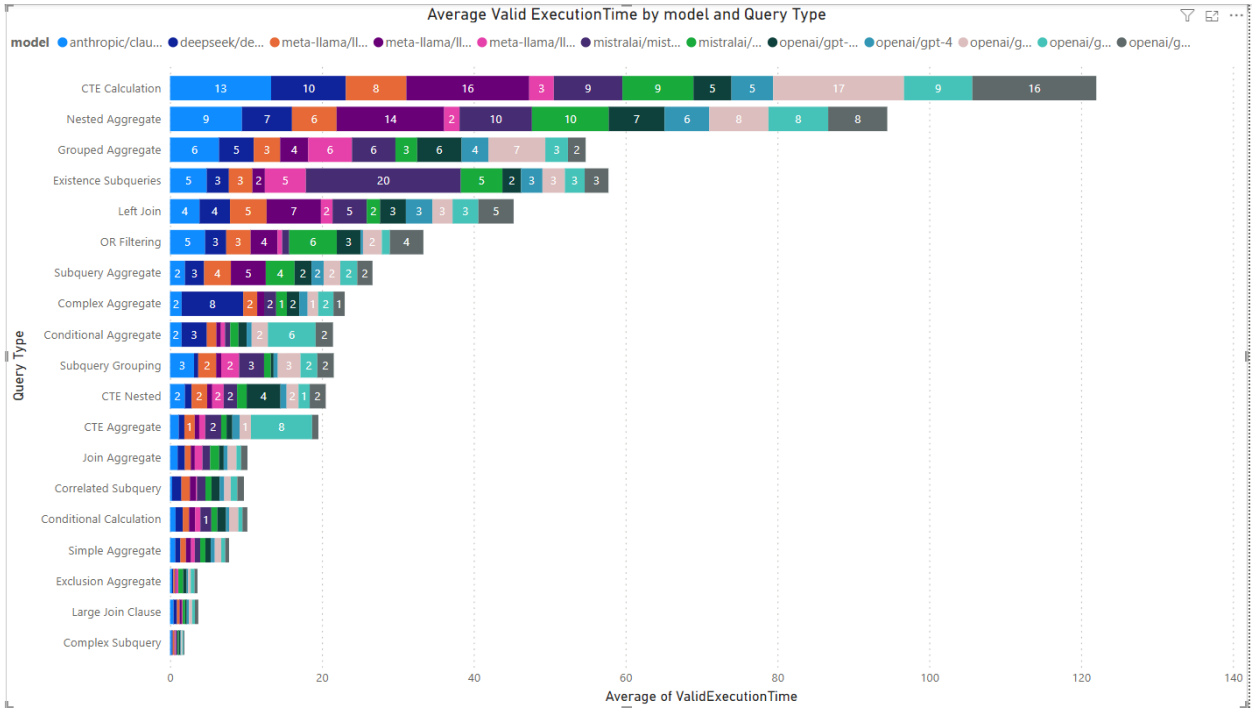
In order to generate the LLM responses to be evaluated, we used a combination of Python and a service called OpenRouter. OpenRouter provides an interface to use multiple large language models and compare their responses. Through Python we used the OpenRouter API to automate through different prompts for each LLM. We also used GitHub for version control and code management.

We evaluated query generation responses from a total of 12 different large language models. We used 5 models from OpenAI, 3 models from Meta, 2 models from Mistral AI, 1 model from Deepseek called Deepseek Coder, and the Claude-3.5 Sonnet model from Anthropic. The models from OpenAI were GPT-4 Turbo, GPT-4, GPT-4o, GPT-4o mini, and GPT-3.5. The models from Meta were all Llama-3.1 each with a different number of parameters. The models from Mistral AI were Mistral-7b-instruct and Mistral Large. These models were chosen because they are all developed by prominent AI research organizations and represent cutting edge capabilities within the LLM space.

Before asking the models to generate and SQL queries, they were fed the information about the database schema. There were 4 different preface prompts that were also used before asking any questions. These preface prompts were used to give the models more information about their task in slightly different ways, which was to generate the fastest possible running SQLite within <query> tags. The actual prompts used for query generation were the same throughout all LLMs. We used 22 questions that were derived from the first 4 sections of the TPC-H specifications sheet. These sections are 'Logical Database Design', 'Queries and Refresh Functions', 'ACID Properties' (Atomicity, consistency, isolation, and durability), and finally 'Scaling and Database Population'. When getting results for each query generated, we recorded each queries validity, accuracy and speed.

Results & Analysis

Performance Evaluation of Models Across SQL Query Types

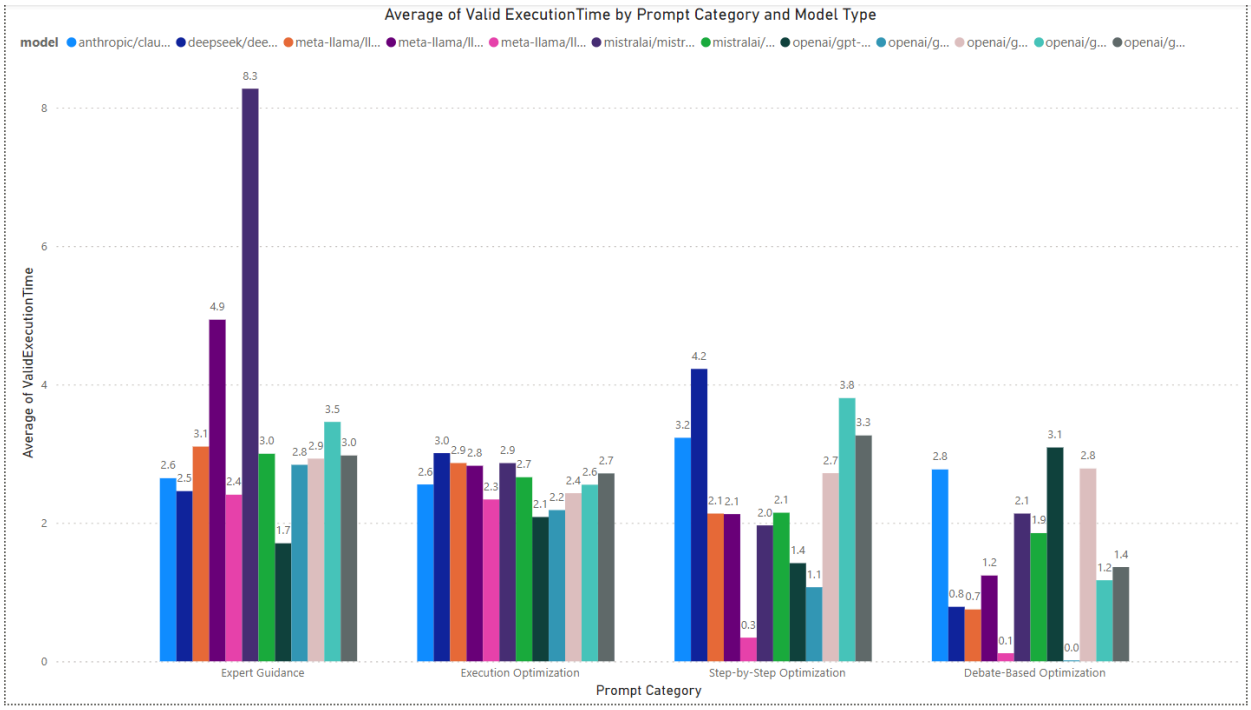


1. **Dominant Query Types**:
    - CTE Calculation and Nested Aggregate queries show the highest average execution times, indicating they are the most computationally intensive among the types evaluated.
    - CTE Calculation has the highest peak, reflecting complex operations that involve multiple calculations across the dataset.
2. **Performance Across Models**:
    - The OpenAI/GPT-4 model demonstrates consistent performance across most query types but still exhibits higher execution times in the more complex queries like CTE Calculation and Nested Aggregate.
    - DeepSeek/DeepSeek-Coder shows some variability but generally handles complex queries like Grouped Aggregate and Existence Subqueries more efficiently compared to others.
    - Meta-Llama models, especially Meta-Llama/Llama-3.1-405b-instruct, exhibit notable peaks in CTE Aggregate and CTE Nested queries, reflecting difficulties with these specific types.
3. **Execution Time Insights**:
    - Across all models, the execution time tends to increase with the complexity of the query.
    - Models exhibit varied efficiency levels, particularly with OR Filtering and Left Join, where some models perform better than others.
4. **Consistency and Outliers**:
    - The Grouped Aggregate and Existence Subqueries show more spread in the execution times across models, indicating inconsistency in handling these queries.
    - Some models, like Mistralai/Mistral-Large, display relatively consistent performance across multiple query types, except in the more complex categories.
5. **Comparative Analysis**:

- Anthropic/Claude-3.5-Sonnet appears to handle simpler query types such as Simple Aggregate efficiently but struggles with more advanced query types, particularly those involving Nested Aggregates.
- OpenAI/GPT-4-Turbo demonstrates slightly better performance on average, particularly with CTE Aggregate and Join Aggregate queries, compared to other models.

This analysis highlights the varying capabilities of different models in handling specific SQL query types. The overall trend suggests that as query complexity increases, execution time escalates, with some models managing this better than others. The insights provided can guide the selection of models based on the type of query workload anticipated in practical applications.

Comparison of Execution Time by Model and Prompt Category

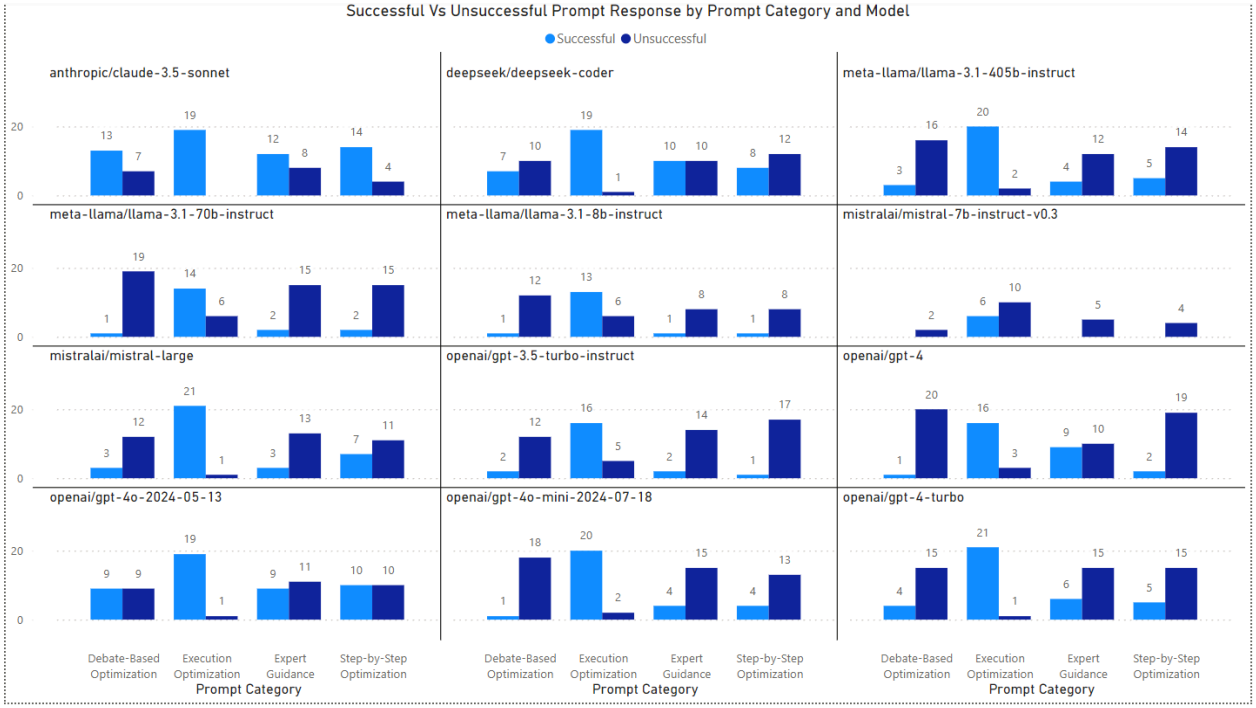


1. **DeepSeek/Coder with Expert Guidance:**
    a. Shows the highest execution time of 8.3 seconds.
    b. Indicates potential inefficiency with complex prompts in this category.
2. **Mistral/Large with Execution Optimization:**
    a. Achieves a low execution time of 1.7 seconds.
    b. Demonstrates strong performance in this optimization approach.
3. **GPT-4 Models with Step-by-Step Optimization:**
    a. Displays moderate execution times, around 2.7 to 4.2 seconds.
    b. Consistent performance across models in handling step-by-step prompts.
4. **Meta-LLaMA Models with Debate-Based Optimization:**
    a. Execution times are fairly low, ranging between 1.2 and 3.1 seconds.
    b. Effectively manages debate-based prompts, indicating robust processing capabilities.
5. **General Trend:**
    a. Execution times vary significantly across prompt categories, suggesting that model efficiency is highly dependent on the type of optimization used.

b. Models generally perform best with Execution Optimization and Debate-Based Optimization, with some exceptions like DeepSeek/Coder in Expert Guidance.

This analysis highlights the variability in execution times across different models and prompt categories, emphasizing the need to select the appropriate model and optimization strategy for specific tasks.

Comparison of Successful vs Unsuccessful Prompt Responses by Model and Prompt Category



1. **DeepSeek/Coder Performance:**
    a. Excels in Execution Optimization with 19 successful responses.
    b. Struggles with Debate-Based Optimization, having 7 successful vs. 7 unsuccessful responses.
2. **Meta-LLaMA/LLaMA-70B-Instruct:**
    a. Outstanding in Execution Optimization, with 19 successful vs. 1 unsuccessful.
    b. Consistently low success across other categories, indicating a specialization in execution.
3. **OpenAI/GPT-4 Performance:**
    a. Shows balanced performance across categories, with higher success in Execution Optimization and Step-by-Step Optimization.
    b. Notably, 19 successful responses in Step-by-Step Optimization.
4. **Anthropic/Claude-3.5-Sonnet:**
    a. Performs reasonably well across all categories.
    b. Highest success in Execution Optimization but lower in other areas compared to other models.
5. **Mistralai/Mistral-7B-Instruct-v0.3:**
    a. Moderate performance across categories, with Execution Optimization showing more successful than unsuccessful responses.
    b. Lacks strong performance in other categories.
6. **General Insights:**

a. Execution Optimization emerges as a strength across models, particularly for DeepSeek, Meta-LLaMA, and OpenAI models.
b. Debate-Based Optimization shows more variability in success, suggesting it may be more challenging for some models.
c. Step-by-Step Optimization and Expert Guidance also show balanced success, especially in models like GPT-4.

This analysis highlights the strengths and weaknesses of different models across various prompt categories, providing valuable insights for selecting the appropriate model for specific tasks.

Overall, the analysis underscores the importance of selecting models based on the specific demands of the task, with OpenAI's models generally excelling in complex queries and a more consistent performance across prompt categories, while others may need targeted improvements in certain areas.

Conclusion

In conclusion, as we explore the potential of Large Language Models (LLMs), in generating SQL queries for database management, we find ourselves at the intersection of traditional methods and cutting-edge AI technology. While LLMs offer promising capabilities, including the ability to automate and potentially decrease the time spent on the process of managing large and complex datasets, their effectiveness still needs to be carefully looked into.

Comparing LLM-generated queries to those written by experienced database professionals will allow us to understand whether these AI models can consistently match or even surpass human expertise in accuracy, efficiency, and resources optimization. If LLMs prove to be reliable and efficient, they could transform database management by reducing the time and effort required for query writing, ultimately leading to cost savings for organizations. However, it remains uncertain whether LLMs can fully replace human expertise. While they offer significant benefits, the meticulous understanding, contextual knowledge, and human factors that professionals bring to the table may still be essential in complex or high-stakes scenarios.

Works Cited

Garcia-Molina, H., Ullman, J. D.,, Widom, J. (2009). Database systems - the complete book (2. ed.).. Pearson Education. ISBN: 978-0-13-187325-4

Li, Z., Yuan, H., Wang, H., Cong, G., & Bing, L. (2024). LLM-R2: A Large Language Model Enhanced Rule-based Rewrite System for Boosting Query Efficiency. arXiv. https://arxiv.org/abs/2404.12872

Liu, J., & Mozafari, B. (2024). Query Rewriting via Large Language Models. *arXiv*. https://arxiv.org/abs/2403.09060

Painter, J. L., Chalamalasetti, V. R., Kassekert, R., & Bate, A. (2024). Bridging the gap in drug safety data analysis: Large language models for SQL query generation. https://arxiv.org/abs/2406.10690

Pal, S. (2024, April 9). The WHO PIDM in focus - Building a global community. World Health Organization. Retrieved from https://www.who.int/teams/regulation-prequalification/regulation-and-safety/pharmacovigilance

Schulhoff, S., Ilie, M., Balepur, N., Kahadze, K., Liu, A., Si, C., Li, Y., Gupta, A., Han, H., Schulhoff, S., Dulepet, P. S., Vidyadhara, S., Ki, D., Agrawal, S., Pham, C., Kroiz, G., Li, F., Tao, H., Srivastava, A., Da Costa, H., Gupta, S., Rogers, M. L., Goncearenco, I., Sarli, G., Galynker, I., Peskoff, D., Carpuat, M., White, J., Anadkat, S., Hoyle, A., Resnik, P. (2024). The Prompt Report: A Systematic Survey of Prompting Techniques. arXiv. https://arxiv.org/abs/2406.06608

Supplemental Resources

- Github Repository with data and code