ESP-JETA
ESP Journal of Engineering & Technology Advancements
ISSN: 2583-2646 / Volume 5 Issue 1 January 2025 / Page No: 6-11
Paper Id: JETA-V5I1P102 / Doi: 10.56472/25832646/JETA-V5I1P102

Original Article

Leveraging Large Language Models for Intelligent Data Source Selection and Query Generation in Multi-Database Systems

Nikunj Agarwal
[1]*Computer Science, Visvesvaraya Technological University, Karnataka, India / Senior Software Engineer at Amazon USA.*

Received Date: 10 November 2024 Revised Date: 18 December 2024 Accepted Date: 05 January 2025

Abstract: *The exponential growth of data across industries has led to the emergence of complex, multi-database systems, necessitating intelligent and efficient methods for data source selection and query generation. This research explores the transformative potential of Large Language Models (LLMs) in addressing these challenges. By leveraging their advanced natural language understanding and contextual reasoning capabilities, LLMs can dynamically select relevant data sources and generate optimized queries tailored to specific user inquiries and operational contexts. We propose a framework that integrates LLMs to streamline data retrieval and enhance decision-making processes across multiple domains. Our approach demonstrates its applicability in building efficient chatbots and intelligent systems that provide real-time, accurate, and context-aware responses. Additionally, this system ensures adherence to domain-specific rules and regulations while optimizing performance in handling diverse and distributed data environments. The findings highlight the versatility and efficiency of LLM-powered solutions in revolutionizing data-driven workflows across industries.*

Keywords: *Contextual Query Generation, Machine Learning for Databases, Multi-Database Query Optimization, Prompt Engineering, Large Language Models, Natural Language Processing.*

I. INTRODUCTION

The rapid proliferation of data across industries has led to increasingly complex multi-database systems, posing significant challenges in effectively utilizing and managing diverse data sources. Traditional approaches to data integration and query generation, such as rule-based systems and database management tools, have proven effective for specific use cases but fall short in dynamic, large-scale, and heterogeneous environments. These methods often struggle with limitations such as static query logic, lack of contextual understanding, and difficulty in adapting to evolving user needs.

The advent of Large Language Models (LLMs) like GPT-4, combined with platforms such as Amazon Bedrock, Google Vertex AI, and Microsoft Azure Cognitive Services, has revolutionized how organizations can interact with and extract value from distributed data sources. LLMs excel in understanding natural language inputs, extracting semantic intent, and dynamically generating queries tailored to specific data structures and contexts. This capability is especially crucial for multi-database systems where seamless access to disparate data sources is essential for informed decision-making and real-time analytics.

Each platform offers unique strengths in applying LLMs to multi-database systems. Google Vertex AI, for instance, is known for its advanced conversational capabilities and ability to maintain coherence across long interactions. Similarly, Microsoft Azure Cognitive Services provides robust tools for enterprise-grade virtual assistants. This research, however, focuses on leveraging Amazon Bedrock's seamless integration with AWS infrastructure to build intelligent systems for data source selection and optimized query generation. Amazon Bedrock's flexibility and scalability make it an ideal choice for creating innovative solutions that address the challenges of multi-database environments.

This paper proposes a novel framework that utilizes LLMs to dynamically identify relevant data sources and generate efficient queries in multi-database systems. The framework's application spans industries, offering the ability to build intelligent chatbots and systems capable of providing real-time, context-aware responses. By combining the contextual reasoning of LLMs with domain-specific knowledge bases, this framework ensures not only precision and efficiency but also compliance with industry-specific rules and regulations. This integration aims to transform how businesses interact with complex data landscapes, enhancing operational efficiency and decision-making across multiple domains.

II. SERVICE DESIGN AND IMPLEMENTATION

The proposed implementation focuses on creating an intelligent, responsive web service that integrates Large Language Models (LLMs) with Retrieval Augmented Generation (RAG) workflows to optimize data source selection and dynamic query generation. The core product is built to handle heterogeneous data systems and is designed with scalability and extensibility in mind.

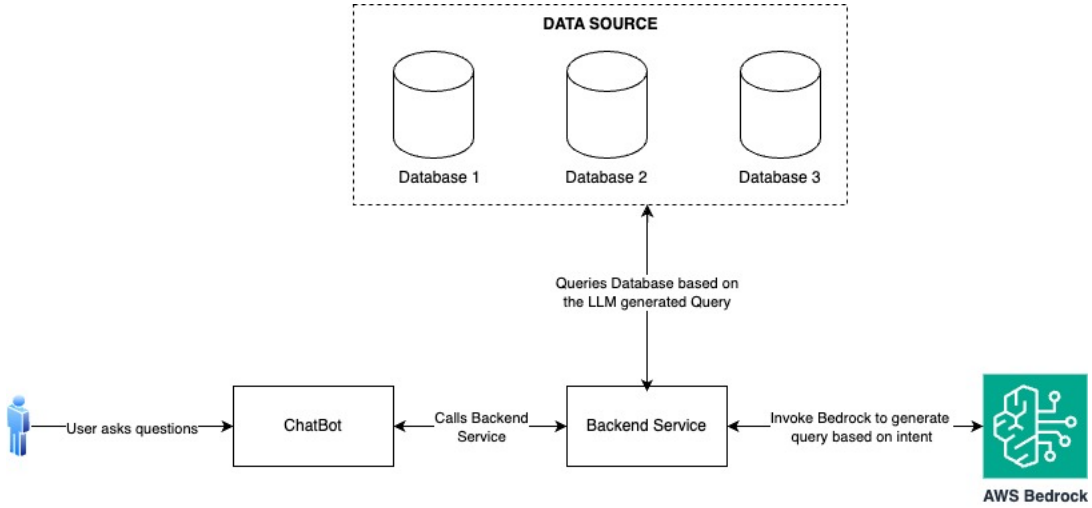

Figure 1: High Level System Design

A. System Architecture

a) Frontend

The user interface is developed using **React** to ensure a clean, responsive, and intuitive experience. The interface allows users to input natural language queries and provides real-time feedback through a chatbot-like interaction.

b) Backend

The backend is powered by Amazon Bedrock, which integrates LLM capabilities with a robust retrieval system. Bedrock serves as the engine for dynamic query generation, data retrieval, and context-aware decision-making.

c) Data Source

The system connects to multiple structured and unstructured data sources, such as relational databases (e.g., PostgreSQL, Redshift), document-based databases (e.g., DynamoDB, MongoDB), and knowledge bases designed to store metadata and domain-specific rules.

B. Core Implementation Components

a) Pre-Processing:

i)User Input:

The system takes unstructured input (e.g., free-text queries) and processes it through natural language preprocessing pipelines.

Steps:

- Tokenization and normalization of user queries.
- Context extraction to identify entities, intent, and relationships within the query.
- Mapping extracted entities to data schemas.

ii) Schema Matching:

If the query references multiple data sources, the system leverages schema metadata to determine the most relevant databases and tables to query.

b) Orchestration and Query Generation:

i) Knowledge Base Integration

Amazon Bedrock's Knowledge Base capabilities play a central role in guiding query generation.

1. Capabilities:
   - Context-Aware Data Selection: The system uses user intent to dynamically identify the most relevant data sources.
   - Rule-Based Reasoning: Predefined domain rules ensure compliance and relevance.

- Source Attribution: The system tracks and attributes each recommendation or query result to its originating source.

ii) Dynamic Querying via RAG Workflow:

- Input Processing: Bedrock's LLM normalizes the user query and classifies it based on pre-trained embeddings.
- Retrieval: Relevant data chunks are fetched using vector similarity search or direct queries from connected databases.
- Generation: The LLM synthesizes retrieved information to generate a precise, natural language response or an optimized query.

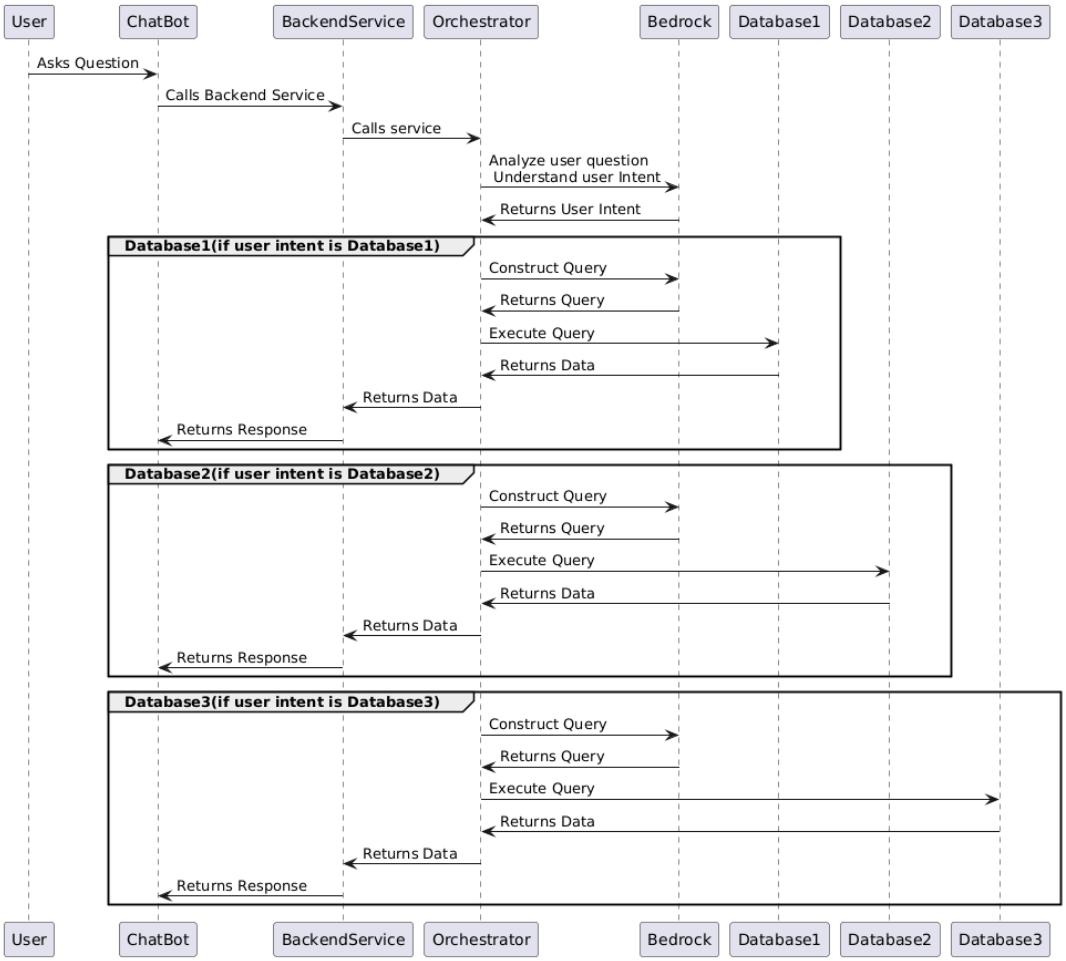

Figure 2: Implementation Flow Chart

c) Post-Processing:

After query execution, the system performs:

- Data Formatting: Ensures consistency in outputs, such as tables, charts, or natural language summaries.
- Contextual Enhancement: Adds explanations or justifications for the query results.
- Feedback Loop: Logs user interactions to improve future responses through fine-tuning.

C. Amazon Bedrock Implementation

The choice of Amazon Bedrock is driven by its advanced LLM capabilities and seamless AWS ecosystem integration. Bedrock simplifies the complexity of building an intelligent query generation system through the following:

- Managed LLM Workflow: Bedrock eliminates the need for setting up and maintaining custom infrastructure for natural language processing and generation.
- Dynamic Query Support: The platform's natural language query capabilities remove the requirement for complex backend databases like Pinecone or other vector databases.
- Scalability: Bedrock's ability to handle high query loads and scale dynamically fits well with multi-database environments.

D. Real-World Example

- User Query: A user asks, *"Which data sources are most relevant for analysing customer churn?"*
- System Processing:
  - Input Normalization: The query is pre-processed to extract key terms: *data sources*, *customer churn*.
  - Knowledge Base Reference: Bedrock consults rules that define customer churn-related metrics and relevant data sources.
  - Dynamic Query Generation: The system identifies tables in databases containing customer engagement and transaction data.
- Response: The system returns a prioritized list of relevant data sources, along with a recommended SQL query to extract churn-related metrics.

E. Human-Centred Design:

While the system automates data source selection and query generation, human oversight is critical. Domain experts—such as data scientists and product managers—design the rules and maintain the knowledge base to ensure alignment with real-world scenarios and ethical standards.

III. PROMPTING TECHNIQUES USED

Prompting is a critical technique in Large Language Models (LLMs), serving as a mechanism to guide the model's responses by providing structured input examples. It leverages the model's pretrained knowledge to generate accurate and context-aware outputs. Several prompting techniques exist, each tailored for specific use cases and varying levels of complexity. Below, we compare Schema-Aware, Few-Shot, and Zero-Shot prompting techniques and their relevance in applications like query generation and intelligent data source selection.

- Zero-Shot Prompting: Zero-shot prompting involves asking the model to perform a task without providing any prior examples. While this approach is useful for straightforward queries, it can lack precision in complex or nuanced tasks.
- Few-Shot Prompting: Few-shot prompting provides the model with a small number of examples to illustrate the desired response format. This technique enhances the model's understanding of the task and improves output accuracy compared to zero-shot prompting.
- Schema-Aware Prompting: Schema-aware prompting embeds schema information directly into the prompt, ensuring the model understands the structural relationships between tables or databases. This technique is particularly tailored for complex multi-database systems, reducing the likelihood of errors in query generation.

A. Importance of Schema-Aware Prompting

Schema-aware prompting is essential for query generation in multi-database systems because it enables:

- Schema Compliance: Ensures queries align with the structural relationships in the database.
- Cross-Database Querying: Facilitates understanding of relationships between schemas in different databases.
- Error Reduction: Reduces the likelihood of syntactical or logical errors by embedding relevant metadata.

B. Example: Intelligent Query Generation Using Schema-Aware Prompting

For a multi-database system, schema-aware prompting can significantly enhance query generation by embedding metadata and relationships in the prompt. For instance:

a) Scenario:

A user requests to find customers from the USA who placed orders exceeding $100.

b) Schema Metadata:

- Database 1: Customers (CustomerID, Name, Country)
- Database 2: Orders (OrderID, CustomerID, OrderDate, Amount)

c) Query:

- SELECT Name
- FROM Customers
- JOIN Orders ON Customers.CustomerID = Orders.CustomerID
- WHERE Country = 'USA' AND Amount > 100;

This approach ensures schema awareness and logical accuracy, especially for cross-database interactions.

C. Techniques in Schema-Aware Prompting

- Schema Embedding: Embed the schema information (tables, columns, relationships, and constraints) directly into the prompt to help the model understand the database structure.

- Relationship Contextualization: Highlight the relationships between tables (e.g., primary keys, foreign keys) to help the model generate queries that respect these connections.
- Natural Language Mapping: Convert the schema details into natural language descriptions that align with the user's query for better comprehension.
- Schema-Driven Constraints: Include schema constraints, such as data types, ranges, or unique attributes, to help the model respect database rules.
- Multi-Schema Contextualization: For multi-database systems, explicitly define how schemas relate across databases and include metadata to inform the model.
- Query Examples with Schema: Provide a few examples of queries based on the schema to demonstrate the expected output format.
- Dynamic Schema Adaptation: Dynamically adjust the schema details included in the prompt based on the user's query, showing only relevant tables and columns.

By utilizing schema-aware prompting, the system establishes a reliable and structured approach to query generation, seamlessly integrating database schemas to improve precision, ensure consistency, and efficiently navigate complex multi-database scenarios.

D. Comparison Summary

Table 1: Comparison of Prompting Technique

Technique	Advantages	Limitations	Use Case
Zero-Shot Prompting	Simple, efficient for straightforward tasks.	Struggles with complex or nuanced queries.	Identifying relevant data sources or basic query generation.
Few-Shot Prompting	Improves accuracy and sets clear expectations.	Requires manual curation of examples.	Generating SQL queries with joins, aggregations, and filtering.
Schema-Aware Prompting	Ensures schema compliance and cross-database accuracy.	Increases prompt complexity and size.	Complex queries in multi-database systems with schema relationships.

IV. COMPARISON OF LARGE LANGUAGE MODELS

The selection of a suitable Large Language Model (LLM) for intelligent data source selection and query generation in multi-database systems requires evaluating the unique features and strengths of various models. Prominent LLMs used in this domain include OpenAI's GPT series, Google's Gemini, Anthropic's Claude, and Amazon's Titan models available via Bedrock. Each model offers distinct advantages tailored to specific needs:

A. OpenAI's GPT Series:

Known for its adaptability and advanced natural language understanding, the GPT series is highly effective in generating structured and unstructured queries, enabling seamless interaction with multi-database systems. While it excels in handling diverse inputs and complex queries, its computational requirements may pose challenges for large-scale, resource-constrained deployments.

B. Facebook's LLaMA:

LLaMA (Large Language Model Meta AI) is known for its focus on accessibility and efficiency. It is designed to be lightweight and can be deployed in environments with resource limitations. LLaMA provides strong capabilities in handling structured queries and is particularly well-suited for applications requiring flexibility and cost-efficiency. While it is not as powerful as some other models in generating highly creative content, its efficiency and ability to handle large-scale queries in multi-database systems make it a competitive choice in resource-constrained environments.

C. Anthropic's Claude:

With a focus on safety and responsible AI, Claude is a top contender for applications requiring high accuracy and contextual understanding, such as selecting relevant data sources in critical industries like finance and healthcare. It offers robust performance in sensitive environments but may have fewer optimization tools for integration into specific database systems.

D. Google's Gemini:

Gemini's strength lies in its contextual awareness and ability to manage intricate, domain-specific queries. Its multilingual support and integration into Google's ecosystem make it ideal for organizations prioritizing conversational interfaces or global data queries. However, its dependence on Google's infrastructure might limit flexibility for independent enterprise solutions.

E. Amazon's Titan Models via Bedrock:

Titan models, available through Amazon Bedrock, excel in large-scale, schema-aware query generation and enterprise-grade integration. Their seamless compatibility with AWS services and optimized support for retrieval-augmented workflows make them ideal for handling structured and unstructured queries across complex multi-database architectures.

These models vary in their scalability, integration options, and suitability for specific use cases. While OpenAI's GPT offers unmatched generative capabilities, Gemini is highly effective for maintaining coherence in conversational data selection. Anthropic's Claude prioritizes safety and precision, making it ideal for sensitive applications, whereas Amazon's Titan models stand out for their enterprise scalability and efficient schema-aware query generation. Facebook's LLaMA, on the other hand, excels in resource efficiency, offering a balance of accessibility and power for resource-conscious deployments. The choice of an LLM depends on factors such as deployment environment, integration requirements, and the complexity of the query scenarios.

V. CONCLUSION

The integration of Large Language Models (LLMs) into multi-database systems marks a transformative shift in how data is sourced, queried, and utilized across industries. By leveraging platforms such as Amazon Bedrock and integrating knowledge bases tailored to domain-specific requirements, this study has demonstrated how LLMs can optimize data retrieval processes, ensuring real-time, context-aware decision-making. The use of techniques like schema-aware prompting and retrieval-augmented generation has enhanced the accuracy and efficiency of query generation, enabling systems to seamlessly navigate complex, heterogeneous data environments.

While the findings presented in this paper represent a significant advancement in data management, the rapidly evolving landscape of AI and LLM technologies offers exciting opportunities for future research. Future work should focus on further enhancing the scalability of these systems, improving their ability to handle even more diverse and dynamic data sources, and refining their integration with various data infrastructures. Additionally, incorporating more sophisticated user feedback mechanisms and extending capabilities to handle multilingual and cross-cultural contexts will ensure that LLM-powered systems remain adaptable and highly effective in a globalized data landscape. Such advancements will solidify the role of LLMs in revolutionizing data-driven workflows, enabling intelligent systems that exceed the needs of users across industries.

VI. REFERENCES

[1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. A., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 5998-6008.

[2] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., & others. (2020). Language models are few-shot learners. *Proceedings of the 34th International Conference on Neural Information Processing Systems*, 1598-1610.

[3] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of NAACL-HLT*, 4171–4186.

[4] Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training. *OpenAI Blog*.

[5] Petroni, F., Ruder, S., Schmitz, J., & Ruder, S. (2019). Language models as knowledge bases? *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 4299-4308.

[6] Gupta, S., Yadav, A., & Bansal, M. (2021). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL)*.

[7] Amazon Web Services (2023). Amazon Bedrock: Building Generative AI Applications with Foundation Models. *AWS Whitepaper*.

[8] Chen, M., & Zhang, Z. (2020). A Survey on Data Integration and Query Generation Techniques. *International Journal of Database Management Systems (IJDMS)*, 12(6), 75-92.

[9] Zhou, L., & Wang, X. (2021). Multi-Database Query Generation for Data Integration using LLMs. *Journal of Information Systems*, 25(3), 123-145.

[10] Bender, E. M., Gebru, T., McMillan-Major, A., & Shmitchell, S. (2021). On the dangers of stochastic parrots: Can language models be too big? *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, 610-623.