American Academic Scientific Research Journal for Engineering, Technology, and Sciences
ISSN (Print) 2313-4410, ISSN (Online) 2313-4402
https://asrjetsjournal.org/index.php/American_Scientific_Journal/index

AI-Augmented Data Modeling: Enhancing Star Schema Design for Modern Analytics

Yegor Koriagin*

Data Architect, Arizent, Boston, 02210, MA, US

Email: yegor.ykconsulting@gmail.com

Abstract

The star schema remains a foundational dimensional modeling approach in business intelligence, valued for its simplicity, performance, and compatibility with OLAP queries. However, manual schema design is labor-intensive and error-prone in large-scale or rapidly evolving data environments. This study investigates the application of Artificial Intelligence (AI), particularly large language models (LLMs), in automating and optimizing star schema generation. Models such as OpenAI's GPT-4, Google Gemini, and Meta's LLaMA 3 were evaluated for their ability to infer schema structures, enforce relational integrity, and enhance semantic alignment. Experimental results demonstrated that AI-assisted modeling can reduce development time by over 80%, while increasing accuracy and consistency. These findings highlight the growing potential of AI in streamlining enterprise data modeling processes.

Keywords: Star Schema; Dimensional Modeling; Artificial Intelligence; Data Warehousing; LLMs; GPT-4; Google Gemini; Meta LLaMA; Automation.

1. Introduction

Dimensional modeling using star schemas [1] is widely adopted in data warehousing and analytical systems due to its clear structure and efficiency in aggregating and querying large datasets. A star schema typically consists of a central fact table surrounded by multiple denormalized dimension tables. While effective, traditional schema design methods demand significant human effort and are prone to inconsistencies-particularly in organizations managing diverse or frequently changing datasets.Recent advancements in AI, especially the rise of LLMs, present new opportunities to automate schema generation and validation [3]. These models exhibit an ability to interpret metadata, classify fields, and generate database schemas in standardized formats. This paper explores how models such as OpenAI GPT-4, Google Gemini, and Meta LLaMA 3 can augment the schema design process, offering both practical benefits and technical insights.

Received: *7/6/2025*
Accepted: *9/6/2025*
Published: *9/16/2025*

---

* *Corresponding author.*

2. Methodology

2.1 AI Capabilities in Schema Modeling

LLMs were evaluated for their ability to perform core modeling tasks, including:

- **Schema Inference [2]:** Identifying dimensions, facts, keys, and relationships from structured data or metadata.
- **Semantic Classification:** Automatically tagging attributes, measures, and identifiers using learned representations.
- **Join Path Suggestion:** Recommending relational links based on contextual understanding of tabular data.
- **Validation Feedback**: Detecting incomplete or incorrect primary-foreign key configurations.

2.2 Output Standardization

To integrate AI outputs into production pipelines, models were prompted to generate schemas in structured formats:

- **SQL DDL:** Used for direct schema implementation in relational databases.
- **JSON/YAML:** Useful for schema registries and configuration-driven pipelines.
- **DBML:** A human-readable format for schema visualization tools.

To integrate AI-generated schemas into production pipelines, outputs must be machine-readable, verifiable, and structured in a consistent format. While LLMs can produce human-readable text, enforcing schema constraints programmatically is essential to prevent structural errors or inconsistencies during database generation. LangChain [7] offers robust tools for structuring LLM output, including the StructuredOutputParser and PydanticOutputParser, which are designed to enforce strict output schemas through parsing and type-checking.

2.2.1. LangChain StructuredOutputParser

The StructuredOutputParser allows developers to define an output schema using natural language or informal rules. It maps raw LLM responses to structured Python dictionaries or JSON-like objects. This parser is especially useful when schema definitions are relatively simple or when fast prototyping is needed.

2.2.2. PydanticOutputParser

For more rigor and type-safety, PydanticOutputParser leverages Pydantic, a Python data validation library. This parser uses Python classes to enforce strong typing and value validation. It is particularly useful when output fields have nested structures or complex validation rules.

This parser validates that each table contains properly typed columns and that primary key flags are properly set. If the LLM produces an output that violates the schema-for example, a column with a missing type-the parser will raise an error, prompting a retry or fallback behavior.

2.2.3. Practical Benefit in AI-Driven Modeling

By combining prompt engineering with structured parsers, LangChain ensures that LLM outputs are not just readable, but actionable. These tools reduce the risk of:

- Incorrect data types or null schema fields
- Misformatted JSON/SQL output
- Inconsistent or incomplete schema definitions
- Manual rework due to ambiguous model responses

This structured approach aligns LLM capabilities with enterprise data engineering standards, enabling safe integration into CI/CD pipelines, modeling platforms, or documentation tools (e.g., dbdiagram.io, dbt, or Airbyte).

2.3 Experimental Setup

OpenAI GPT-4 [6], Google Gemini (2024) [4], and Meta LLaMA 3 (70B) [5] were tested on the same modeling tasks. Prompts included sample metadata, partial datasets, and business use case descriptions. Each model was expected to identify fact and dimension tables, generate a valid star schema, and produce SQL DDL suitable for deployment in PostgreSQL 17.5.

2.4 Datasets

Three datasets were used:

- **Retail Sales Dataset (UCI Repository):** Real-world sales data from a UK-based e-commerce store.
- **Healthcare Claims Dataset (Synthea):** Simulated insurance claims with patient-provider-diagnosis structures.
- **International Census Dataset (US Census Bureau):** Global population statistics, including time series on fertility, mortality, and migration.

2.5 Previous Studies

Efforts to automate schema generation have evolved from rule-based knowledge systems to advanced LLM-driven frameworks. Two representative approaches highlight this progression.SchemaAgent (2025) introduced a multi-agent framework where specialized large language model (LLM) agents collaborate to generate relational database schemas from user requirements [8]. Each agent focuses on distinct subtasks-such as schema inference, relationship mapping, and validation-while additional reflection and inspection mechanisms detect and correct errors in intermediate steps. To benchmark performance, the authors introduced RSchema, a dataset of more than 500 requirements–schema pairs. Results showed SchemaAgent outperforming single-model baselines by improving both accuracy and resilience against error propagation. This work underscores the growing interest in applying distributed LLM systems to schema automation, offering modularity and improved error handling compared to monolithic model prompts.Earlier, the Semantic-Based Star Schema Designer (2021) demonstrated

how knowledge-based reasoning systems could aid in constructing dimensional models [9]. This system leveraged semantic rules to infer star schema components-such as dimensions, hierarchies, and attribute types-even when input metadata was incomplete or inconsistent. Notably, the system could propose attribute names and predict data types in cases where schema elements were partially defined. The approach was especially valuable for novice designers and semi-structured data, offering a foundation for automated schema reasoning prior to the rise of large-scale LLMs.Taken together, these studies illustrate the trajectory of research in schema automation: from rule-driven semantic systems that ensure logical soundness to LLM-powered multi-agent frameworks capable of handling diverse and complex inputs. The present study builds on this continuum by applying state-of-the-art LLMs (GPT-4, Gemini, LLaMA 3) to automate star schema design, with a focus on structured outputs and integration into production pipelines.

3. Results

3.1 Schema Generation Accuracy

Table1

Model	**Retail Sales (%)**	**Healthcare Claims (%)**	**International Census (%)**	**Avg Accuracy**
OpenAI GPT-4	94%	89%	91%	91.3%
Google Gemini	90%	85%	88%	87.7%
Meta LLaMA 3	86%	80%	83%	83%

Evaluation was based on correct identification of schema elements, logical foreign key usage, and alignment with star schema conventions.

3.2 Development Time Reduction

AI-assisted modeling drastically reduced schema development time. Initial schema design was reduced from approximately 18 hours to 2 hours, representing an 89% time savings. Similarly, validation and quality assurance time decreased from 10 hours to 2 hours, an 80% reduction. These time savings contributed to faster project delivery without compromising design quality.

3.3 Observed AI Enhancements

Models demonstrated the ability to suggest missing surrogate keys and appropriate fact-dimension bridge tables. They also flagged overly granular fact fields-such as product attributes-that were more appropriately modeled as dimensions. Furthermore, column classification accuracy improved when AI leveraged contextual cues from metadata, resulting in clearer, more maintainable schema outputs.

4. Discussion

4.1 Model-Specific Performance Differences

The comparative results reveal that not all LLMs handle schema generation tasks equally. GPT-4's superior performance (91.3% accuracy) is likely attributable to its advanced semantic classification abilities, enabling it to more consistently distinguish between measures, attributes, and identifiers. Gemini and LLaMA 3, while still effective, showed weaknesses in attribute classification and occasionally generated overly complex fact tables. These tendencies suggest that careful model selection is critical for organizations planning to integrate LLMs into schema design workflows.

4.2 Nature of Time Savings

The observed reduction in development time-over 80% in schema drafting and 80% in validation-was not uniform across all stages of modeling. LLMs proved especially effective in initial schema drafting, automating repetitive design tasks such as dimension creation, key assignment, and fact table identification. However, more sophisticated aspects, such as encoding business-specific logic and compliance with governance standards, continued to require human intervention. This indicates that LLMs currently function best as accelerators for routine modeling tasks, while experts remain indispensable for context-sensitive refinements.

4.3 Limitations

While the results of this study demonstrate the potential of LLMs in automating star schema design, several important limitations must be acknowledged:

4.3.1 Dataset Scope and Generalizability

The evaluation was conducted on three datasets (retail sales, healthcare claims, and census data). Although these datasets represent different domains, they may not capture the full diversity and complexity of enterprise-scale data warehouses, which often integrate dozens of heterogeneous data sources. The findings may therefore not generalize to multi-domain or cross-industry environments without further validation.

4.3.2 Semantic Fidelity

The models demonstrated strong structural accuracy but weaker performance in capturing nuanced business semantics. For example, fact-dimension boundaries were sometimes correctly inferred at the technical level but misaligned with business rules. Since data warehouses are designed to support business decision-making, semantic misalignments can significantly reduce schema usefulness even when the design is structurally valid.

4.3.3 Reproducibility of Outputs

LLMs are inherently non-deterministic: the same prompt can yield different schema outputs across multiple runs. This lack of reproducibility complicates production integration, particularly in environments where consistent

results are required for CI/CD pipelines or automated deployments.

4.3.4 Scalability and Context Window Constraints

The current generation of LLMs is limited by token context windows, restricting their ability to process very large datasets or schemas with thousands of attributes simultaneously. While chunking or hierarchical prompting strategies can partially address this, scalability remains a significant barrier for applying LLMs to **enterprise-scale data warehouses**.

4.3.5 Evaluation Metrics

The accuracy metric in this study was based on correct identification of schema elements (fact tables, dimension tables, keys, and relationships). However, it did not account for:

- **Query performance** (e.g., OLAP query optimization on generated schemas).
- **Business interpretability** (alignment of schema design with user requirements).
- **Downstream usability** (ease of integration with ETL pipelines, BI tools).

A more comprehensive evaluation framework is needed for practical deployment scenarios.

4.3.6 Integration and Cost Considerations

Although schema development time decreased significantly, this study do not examine the computational costs of using LLMs at scale. Enterprise adoption may be constrained by high API usage fees, latency, or the infrastructure required to fine-tune and maintain these models. Integration into existing data engineering ecosystems (e.g., dbt, Airbyte, or CI/CD workflows) also introduces engineering overhead not captured in the experimental results.

Although these errors were relatively infrequent, they demonstrate that AI-generated schemas should be subjected to systematic review. Potential mitigation strategies include hybrid workflows with automated validation or query-based schema testing to flag inconsistencies prior to deployment.

4.4 Pioneering Contribution

To the best of the author's knowledge, this is the first systematic study to apply LLMs to data warehouse star schema generation. Prior research has addressed schema labeling [2], schema integration through machine learning [3], or knowledge-based reasoning for star schema design [9], but none have demonstrated how LLMs can directly produce production-ready SQL DDL, or DBML for dimensional modeling. This positions the current work as a pioneering effort, laying the foundation for AI-augmented data warehousing and opening opportunities for future research in schema versioning, AI-driven ETL integration, and governance-aware modeling.

4.5 Human-AI Collaboration

Finally, the results reaffirm that LLMs are not replacements for human architects but rather collaborative partners.

Their strength lies in accelerating repetitive, structurally consistent tasks, while human experts ensure that schemas align with nuanced business semantics, compliance requirements, and domain-specific rules. This hybrid model of collaboration-where AI provides speed and consistency, and humans supply context and oversight-emerges as the most practical path forward in enterprise data modeling.

5. Conclusion

The use of large language models in star schema design marks a significant shift in modern data engineering. Tools like GPT-4, Gemini, and LLaMA have proven capable of reducing schema development time by over 80% while improving structural accuracy and semantic clarity. Though not a replacement for expert designers, these models serve as valuable accelerators in the data modeling lifecycle. Future research may explore automated schema versioning, AI-integrated ETL workflows, and real-time feedback based on system performance.

References

[1] Kimball, R., & Ross, M. (2013). *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling* (3rd ed.). Wiley.

[2] Sharma, A., et al. (2021). Applying NLP for Data Schema Labeling. *Journal of Data Engineering*.

[3] Rao, K. (2023). Dimensional Modeling Automation Using Machine Learning. *ACM SIGMOD Posters*.

[4] Google. (2024). Introducing Gemini 1.5. *Technical Overview*.

[5] Meta AI. (2024). LLaMA 3: Open Foundation Models. *Meta Research Release Notes*.

[6] OpenAI. (2023). GPT-4 Technical Report. *OpenAI Documentation*.

[7] Vasilios Mavroudis (2024). LangChain v0.3., Available: https://hal.science/hal-04817573/

[8] Li, X., Zhang, Y., & Chen, H. (2025). *SchemaAgent: Multi-Agent LLM Framework for Relational Schema Generation*. arXiv preprint arXiv:2503.23886.

[9] Ahmed, H., & Mohamed, S. (2021). *Semantic-Based Star Schema Designer: Automating Dimensional Modeling Using Knowledge Rules*. ResearchGate., Available:https://www.researchgate.net/publication/364324920_GENERATING_DATA_WAREHOUSE_SCHEMA