AIDA - ADVANCED INTELLIGENT DATA ASSISTANT

Pavan Alapati , Aagam Shah, Eleena Boppuri, Santosh Govardhan

ADVISOR: DR. KHALID ABOALAYON

COURSE: MSDA3999 Spring 2025

CLARK UNIVERSITY

Table of Contents

AIDA: Advanced Intelligent data Assistant

Transforming Natural Language into Business Insights

Problem Statement

Business users and analysts often struggle to access insights from enterprise databases due to their lack of SQL proficiency, fragmented schema knowledge, and inconsistent access to business rules. This creates a dependency on technical teams for even simple reporting tasks, slows down decision-making, and limits self-service BI adoption.

Furthermore, existing AI assistants for data querying are inconsistent, fail to respect schema and business logic, and often generate invalid or misleading SQL

Overview of AIDA

What is AIDA?

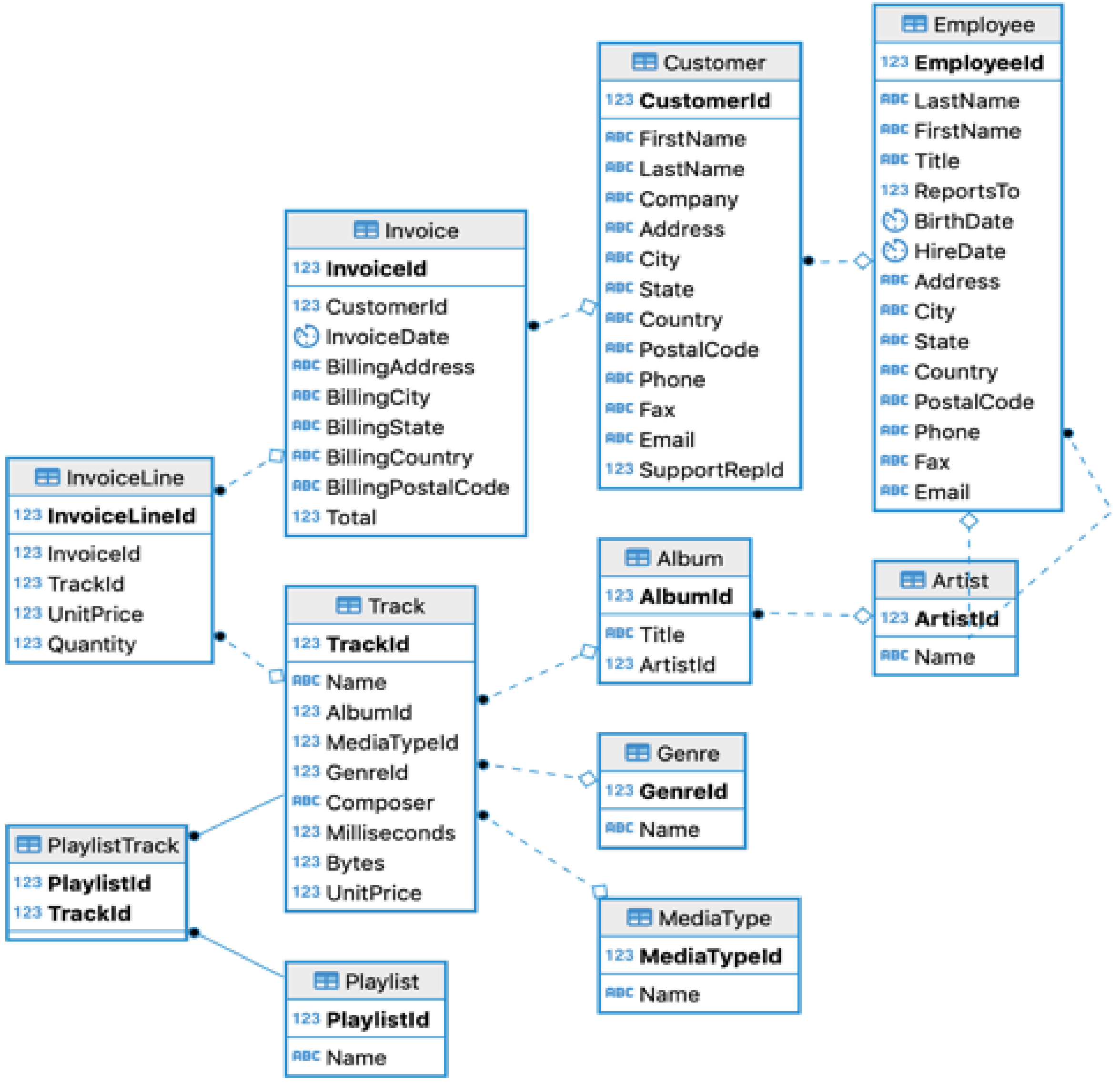
- A modular, LLM-driven assistant for robust NL-to-SQL translations.
- Uses a Retrieval-Augmented Generation (RAG) pipeline.
- Connects Datawarehouse to generate insights (Chinook db)

Key Capabilities:

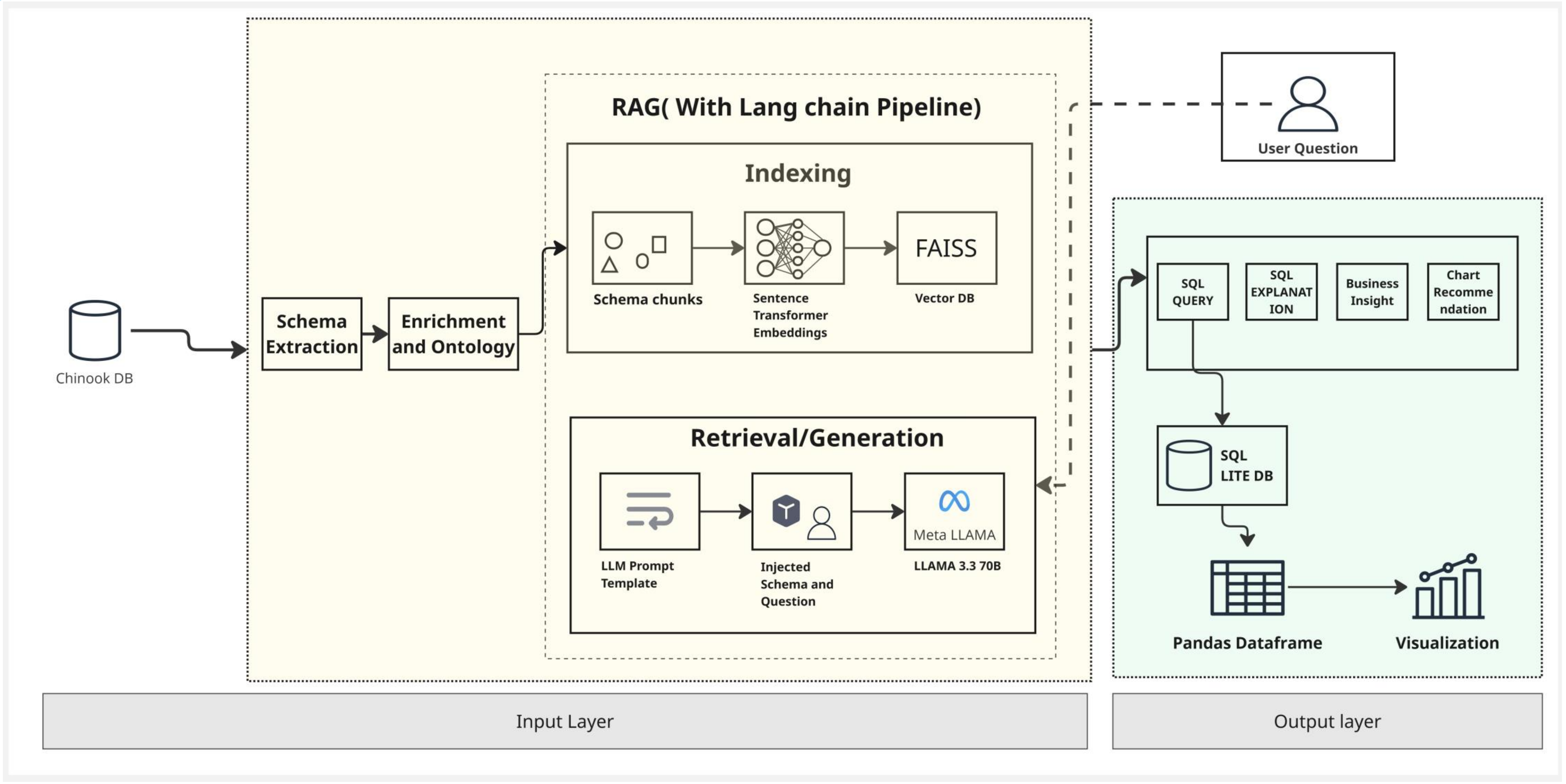
- Schema-aware embeddings
- Business rule indexing
- Structured prompt design
- SQL genration
- Self-healing SQL
- Dynamic Visuals

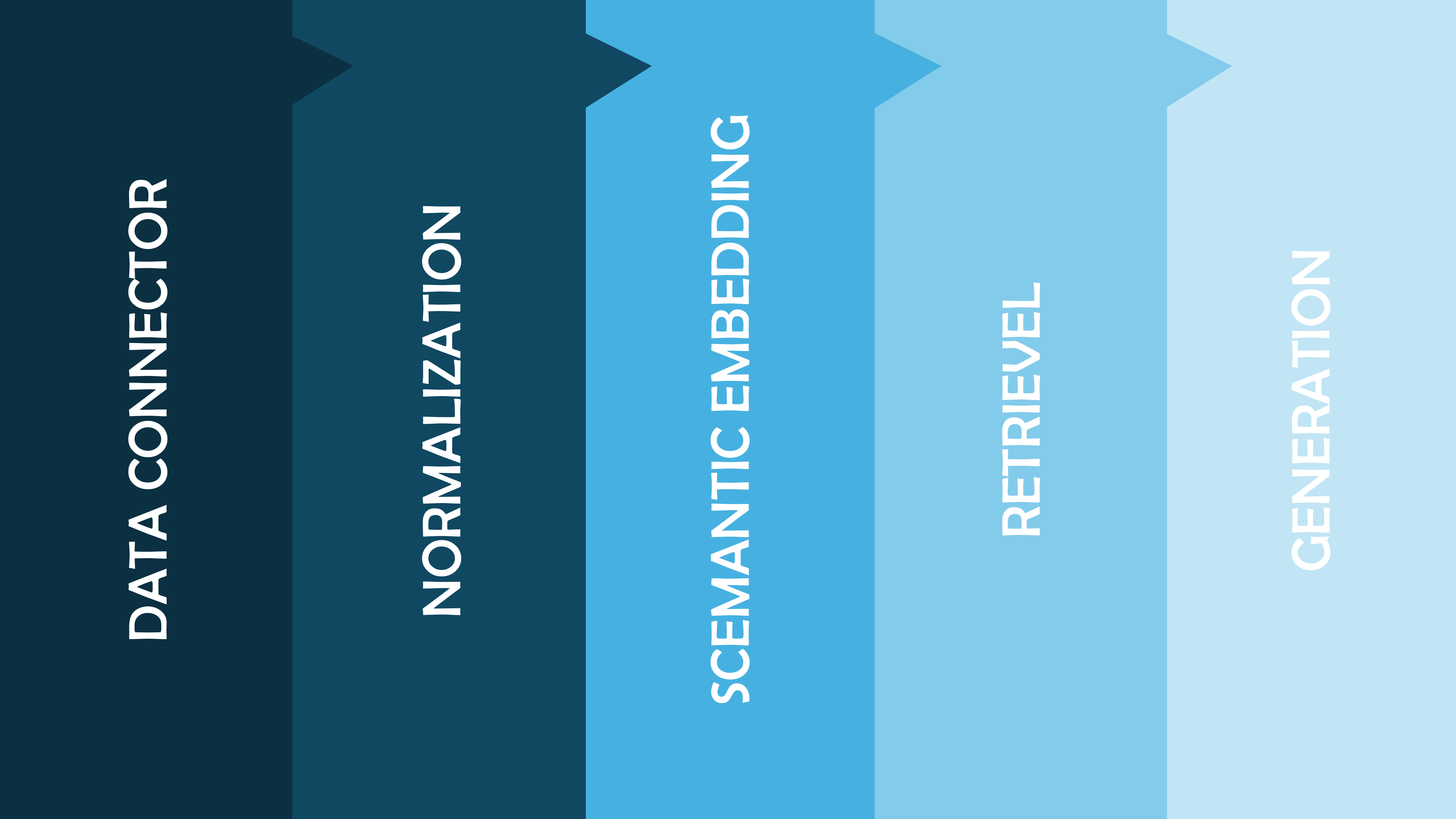
Methodology

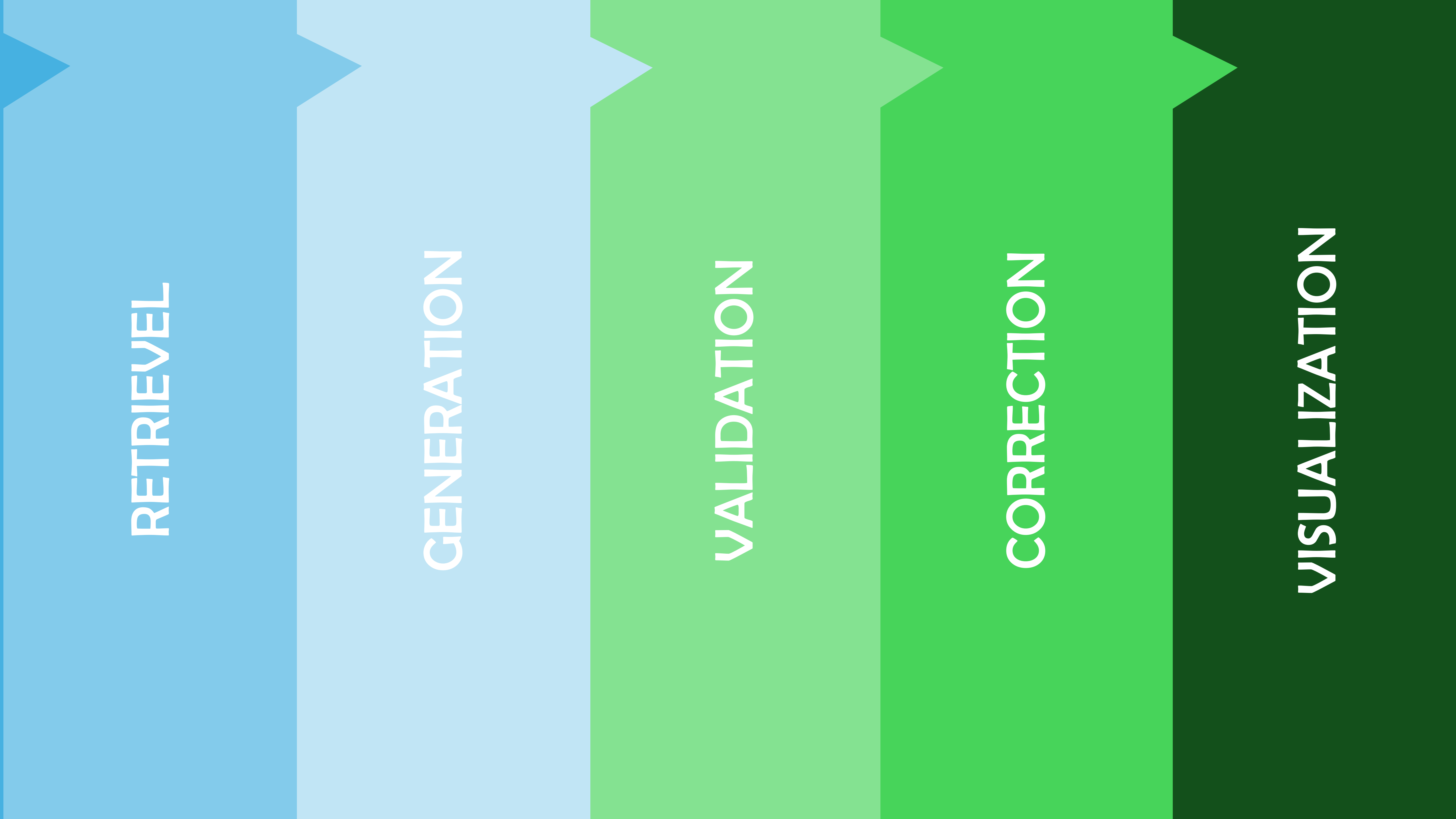
Chinook db ERD diagram



Technical Architecture Overview


Chinook DB
Schema Extraction
Enrichment and Ontology
RAG(With Lang chain Pipeline)
Indexing
Schema chunks
Sentence Transformer Embeddings
FAISS
Vector DB
Retrieval/Generation
LLM Prompt Template
Injected Schema and Question
Meta LLAMA
LLAMA 3.3 70B
User Question
SQL QUERY
SQL EXPLANATION
Business Insight
Chart Recommendation
SQL LITE DB
Pandas Dataframe
Visualization
Input Layer
Output layer


DATA CONNECTOR
NORMALIZATION
SCEMANTIC EMBEDDING
RETRIEVEL
GENERATION


RETRIEVEL
GENERATION
VALIDATION
CORRECTION
VISUALIZATION

DATA CONNECTOR

Query Processing
& Parsing

Input:

- Database (SQLite: Chinook(Music store)

Process:

- Connect and extract schema metadata

Output:

- Raw schema metadata (Tables, Columns, Data Types, Keys)

NORMALIZATION

Normalization

Input:

- Raw schema metadata

Process:

- Convert to structured JSON abstraction
- Semantic enrichment (business-friendly names, descriptions)
- Business term tagging (using dictionaries)

Output:

- Enriched semantic schema JSON

INDEXING

Semantic Embeddings

Input:

- Enriched semantic schema JSON

Process:

- Natural language schema chunking
- Generate embeddings (table-level, column-level) using sentence-transformers
- Index embeddings with FAISS (vector database)

Output:

- Schema embeddings (table-level, column-level)
- FAISS Vector Index

RETRIEVEL

Input:

- Domain-specific business rules, KPIs definitions

Process:

- Chunk business logic into semantic pieces
- Embed chunks with sentence-transformers
- Store embeddings in separate FAISS index

Output:

- Business knowledge embeddings
- Business FAISS Vector Index

Retrieval

GENERATION

Generation

Input:

- User's Natural Language Query
- FAISS Vector Indexes (Schema, Business knowledge)

Process:

- Retrieve most relevant semantic chunks using LangChain retriever
- Hybrid retrieval (table-level, column-level, business knowledge)

Output:

- Contextual semantic chunks

VALIDATION

Validation

Input:

- User's Natural Language Query
- FAISS Vector Indexes (Schema, Business knowledge)

Process:

- Check SQL syntax, Check schema references
- Detect missing joins, misused aliases, or invalid aggregations
- If SQL is valid, If SQL is invalid

Output:

- Validation Status
- Final SQL to pass downstream

CORRECTION

Healing

Input:

- Invalid SQL generated by the LLM
- Error message from SQLite or schema check
- Schema context chunks retrieved via FAISS + LangChain
- Original user question

Process:

- Construct healing prompt
- Send healing prompt to LLM
- Parse LLM response and validate sql again

Output:

- Healed and final sql
- Flag: `was_healed = True/False`
- Healing outcome status `"healed"`, `"healing_failed"`, `"original_executed_successfully"`

VISUALIZATION

Input:

- Validated SQL query results

Process:

- Render visualizations (e.g., Plotly)
- Display via Gradio UI

Output:

- Interactive visual insights and charts

UI Integration

Built with Gradio

Users can:

Ask business questions in plain English

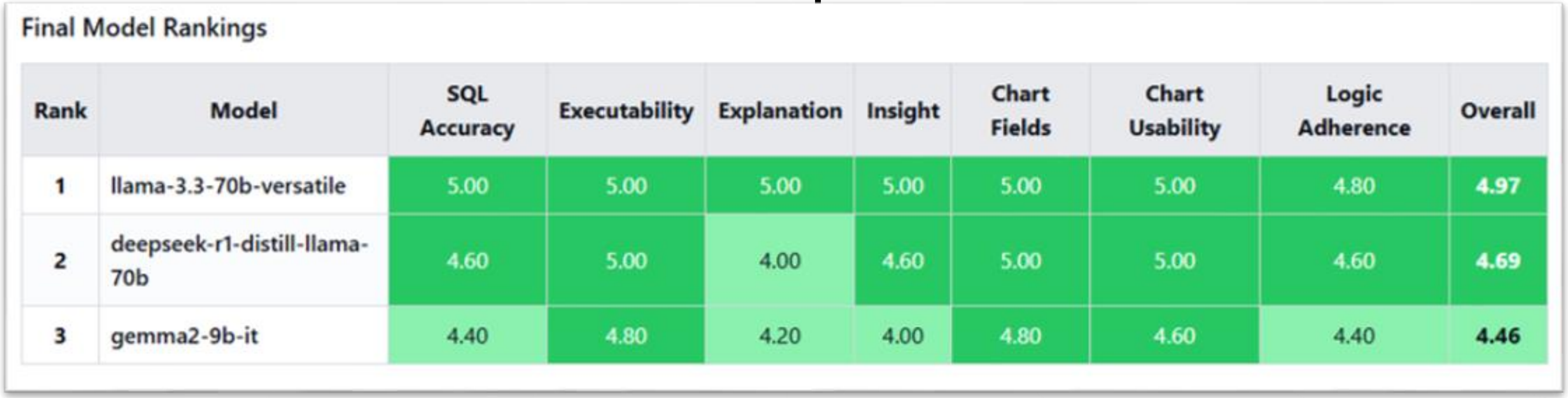
View data visualizations

Read insight explanations

User Flow:

Query → SQL Generation → Results + Explanation → Chart

Model Comparision


Final Model Rankings

Rank	Model	SQL Accuracy	Executability	Explanation	Insight	Chart Fields	Chart Usability	Logic Adherence	Overall
1	llama-3.3-70b-versatile	5.00	5.00	5.00	5.00	5.00	5.00	4.80	**4.97**
2	deepseek-r1-distill-llama-70b	4.60	5.00	4.00	4.60	5.00	5.00	4.60	**4.69**
3	gemma2-9b-it	4.40	4.80	4.20	4.00	4.80	4.60	4.40	**4.46**

- **llama-3.3-70b-versatile:** Exceptional at providing detailed explanations and business insights. Almost always included optimal query structures with appropriate sorting and constraints.
- **deepseek-r1-distill-llama-70b:** Very consistent in SQL accuracy and query executability. Often included helpful column aliases and appropriate join syntax.
- **gemma2-9b-it:** Showed the most improvement throughout the test questions, suggesting good adaptability. Provided clear explanations despite sometimes missing optimizations in earlier questions.

Model Comparision – Possible Reasons for difference

- Parameter Count Scaling Effect
- Architecture Design Differences
- Training Data Composition
- Attention Mechanism Efficiency
- Inference Optimization Tuning
- SQL-Specific Training

Results

- Fully automated NL-to-SQL pipeline with self-healing logic
- Dynamic insights and chart generation from raw data
- Integration of schema and business knowledge using RAG

Challenges

- Maintaining SQL integrity after LLM healing
- Handling ambiguous or generic LLM aliases
- Ensuring consistency across pipeline modules

Lessons learned

- LLMs need tightly structured prompts and guardrails
- Validation and healing logic must preserve semantics
- Modular pipeline design enables debugging and scaling
- Importance of logging user inputs and model outputs to evaluate system behavior, trace errors, and identify patterns in failure cases.

Future Work

- Add explainability and SQL tracing to the UI
- Extend to new datasets with auto-schema discovery
- Introduce conversational memory or multi-turn question handling
- Logging user examples for better retrival results
- Making the system to handle complex business questions

Conclusion

Key Takeaways:

- Scalable Design.
- Human-like Querying.
- Modular & Pluggable.
- Demonstrated capability for enterprise BI applications.
- Foundation for further research and expansion.

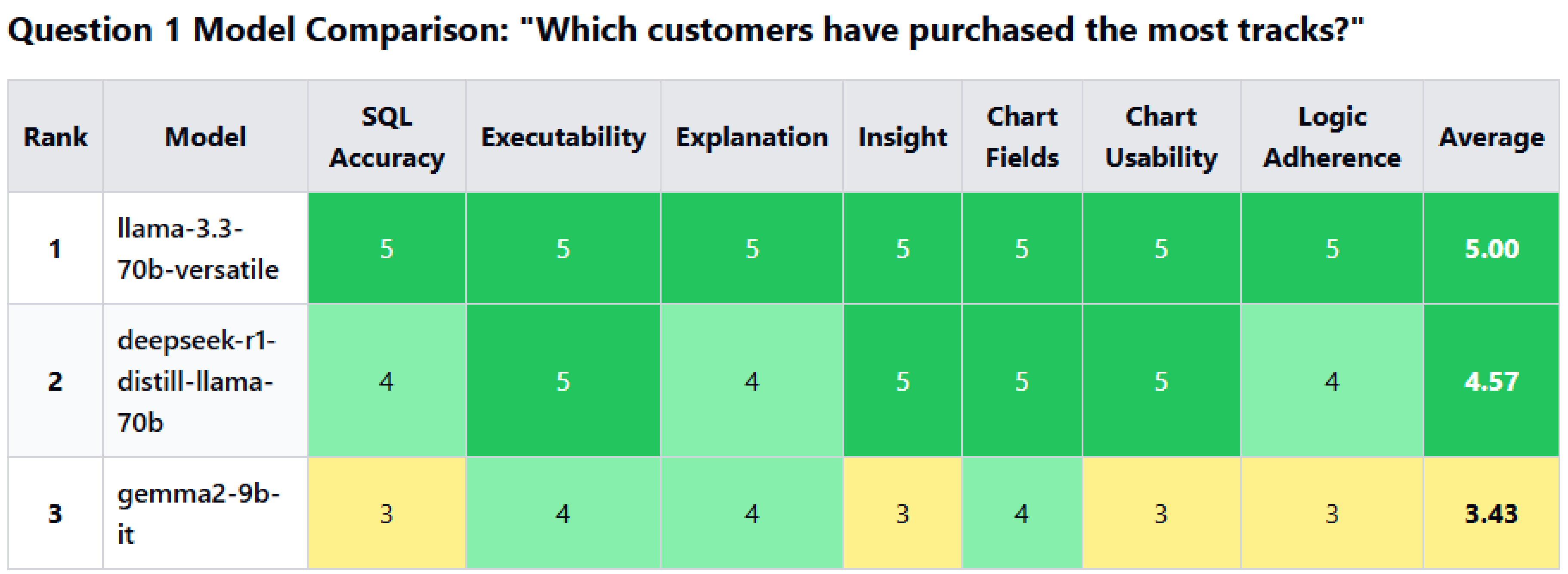
Acknowledgement

We would like to express my sincere gratitude to everyone who supported me throughout the preparation of this presentation. Special thanks to Prof Dr. Khalid Aboalayon and Dr.Ahmed ElSayed,

for their valuable guidance and encouragement. We also appreciate the support and insights from my classmates. Their contributions and feedback have been instrumental in shaping this work.

Questions & Feedback?

References

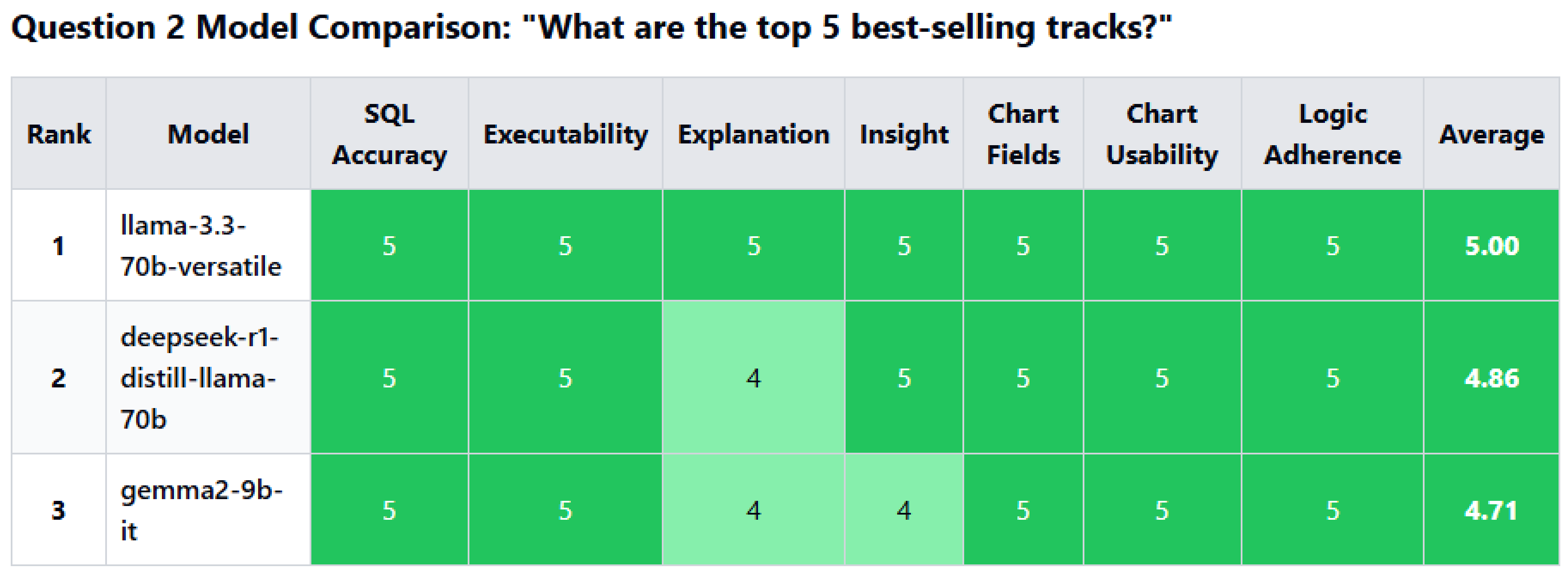

Question 1 Model Comparison: "Which customers have purchased the most tracks?"

Rank	Model	SQL Accuracy	Executability	Explanation	Insight	Chart Fields	Chart Usability	Logic Adherence	Average
1	llama-3.3-70b-versatile	5	5	5	5	5	5	5	**5.00**
2	deepseek-r1-distill-llama-70b	4	5	4	5	5	5	4	**4.57**
3	gemma2-9b-it	3	4	4	3	4	3	3	**3.43**

1st Place: llama-3.3-70b-versatile (5.00) - Perfect scores across all categories, with a comprehensive SQL solution using COUNT(DISTINCT) for accurate track counting and excellent business insights.

2nd Place: deepseek-r1-distill-llama-70b (4.57) - Minor issues in SQL accuracy (using SUM instead of COUNT DISTINCT) but excellent business context and visualization options.

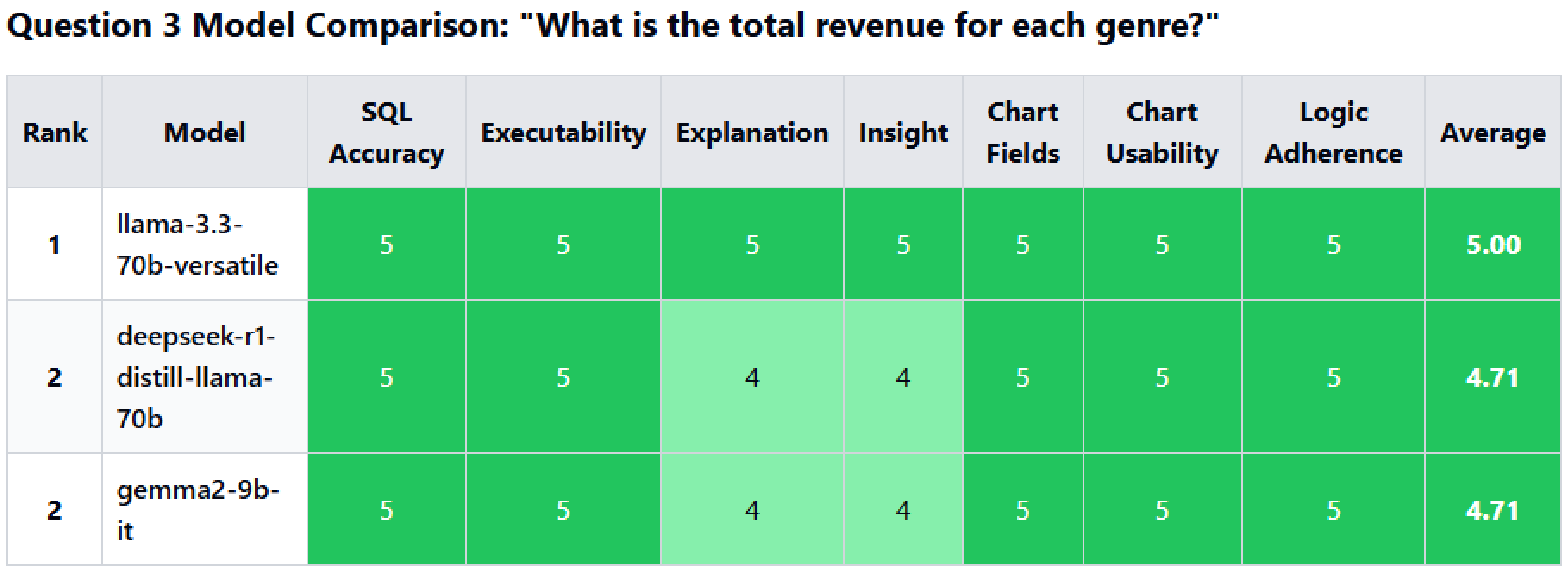
3rd Place: gemma2-9b-it (3.43) - Solution is good but missing customer names in the SELECT clause and potentially counting duplicate tracks without using DISTINCT.



Question 2 Model Comparison: "What are the top 5 best-selling tracks?"

Rank	Model	SQL Accuracy	Executability	Explanation	Insight	Chart Fields	Chart Usability	Logic Adherence	Average
1	llama-3.3-70b-versatile	5	5	5	5	5	5	5	**5.00**
2	deepseek-r1-distill-llama-70b	5	5	4	5	5	5	5	**4.86**
3	gemma2-9b-it	5	5	4	4	5	5	5	**4.71**

Key Observations
1st Place: llama-3.3-70b-versatile (5.00) - Perfect scores across all categories, with a clear SQL query using appropriate joins, grouping, and limiting, coupled with excellent explanations and business insights.
2nd Place: deepseek-r1-distill-llama-70b (4.86) - Nearly perfect solution with very minor issues in explanation depth, but otherwise excellent with appropriate SQL structure and business context.
3rd Place: gemma2-9b-it (4.71) - Strong performance overall with slight shortcomings in explanation depth and insight quality compared to the top models.



Question 3 Model Comparison: "What is the total revenue for each genre?"

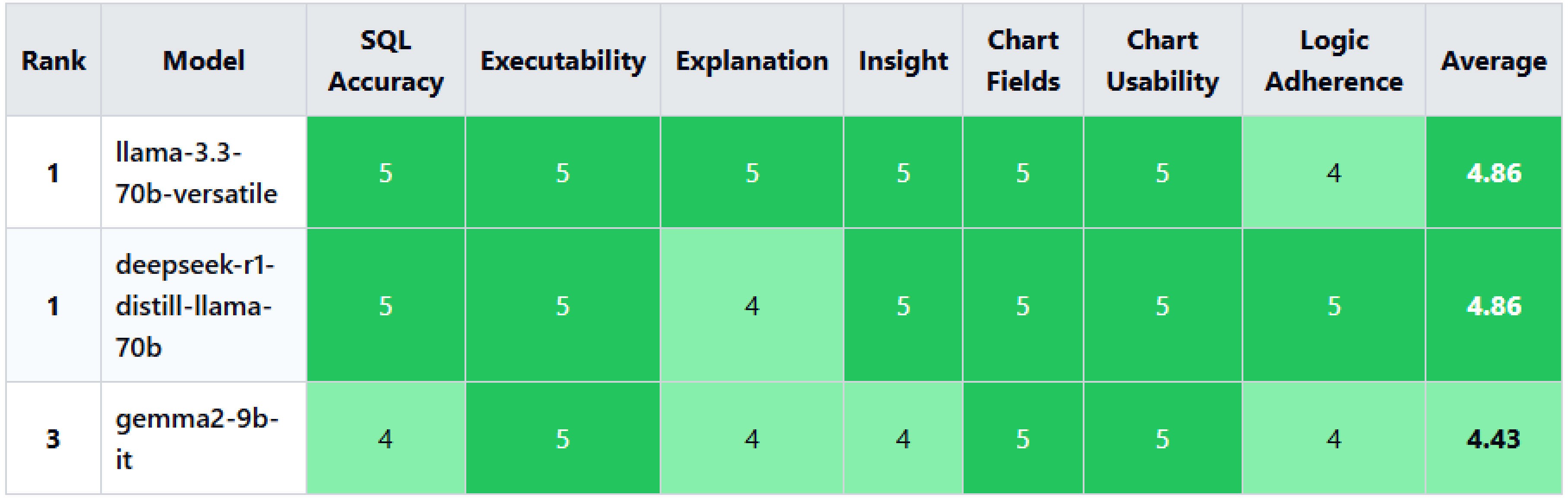
Rank	Model	SQL Accuracy	Executability	Explanation	Insight	Chart Fields	Chart Usability	Logic Adherence	Average
1	llama-3.3-70b-versatile	5	5	5	5	5	5	5	**5.00**
2	deepseek-r1-distill-llama-70b	5	5	4	4	5	5	5	**4.71**
2	gemma2-9b-it	5	5	4	4	5	5	5	**4.71**

Key Observations

1st Place: llama-3.3-70b-versatile (5.00) - Perfect scores across all categories, with detailed explanation that includes units (dollars) and excellent follow-up questions about top performers within genres.

Tied 2nd Place: deepseek-r1-distill-llama-70b and gemma2-9b-it (4.71) - Both models provided highly accurate SQL solutions with minor shortcomings in explanation depth and insight impact compared to llama-3.3-70b-versatile.

Question 4 Model Comparison: "How many tracks are there for each media type?"



Rank	Model	SQL Accuracy	Executability	Explanation	Insight	Chart Fields	Chart Usability	Logic Adherence	Average
1	llama-3.3-70b-versatile	5	5	5	5	5	5	4	**4.86**
1	deepseek-r1-distill-llama-70b	5	5	4	5	5	5	5	**4.86**
3	gemma2-9b-it	4	5	4	4	5	5	4	**4.43**

Key Observations
Tied 1st Place: llama-3.3-70b-versatile and deepseek-r1-distill-llama-70b (4.86) - Both models provided excellent solutions, with llama having slightly better explanations but deepseek adding a useful ORDER BY clause.
3rd Place: gemma2-9b-it (4.43) - Good solution but had a few minor issues: no column aliases in the SELECT statement, no ORDER BY clause, and slightly less comprehensive business insights.

Question 4 Model Comparison: "How many tracks are there for each media type?"

Rank	Model	SQL Accuracy	Executability	Explanation	Insight	Chart Fields	Chart Usability	Logic Adherence	Average
1	llama-3.3-70b-versatile	5	5	5	5	5	5	4	**4.86**
1	deepseek-r1-distill-llama-70b	5	5	4	5	5	5	5	**4.86**
3	gemma2-9b-it	4	5	4	4	5	5	4	**4.43**

Key Observations

Tied 1st Place: llama-3.3-70b-versatile and deepseek-r1-distill-llama-70b (4.86) - Both models provided excellent solutions, with llama having slightly better explanations but deepseek adding a useful ORDER BY clause.

3rd Place: gemma2-9b-it (4.43) - Good solution but had a few minor issues: no column aliases in the SELECT statement, no ORDER BY clause, and slightly less comprehensive business insights.