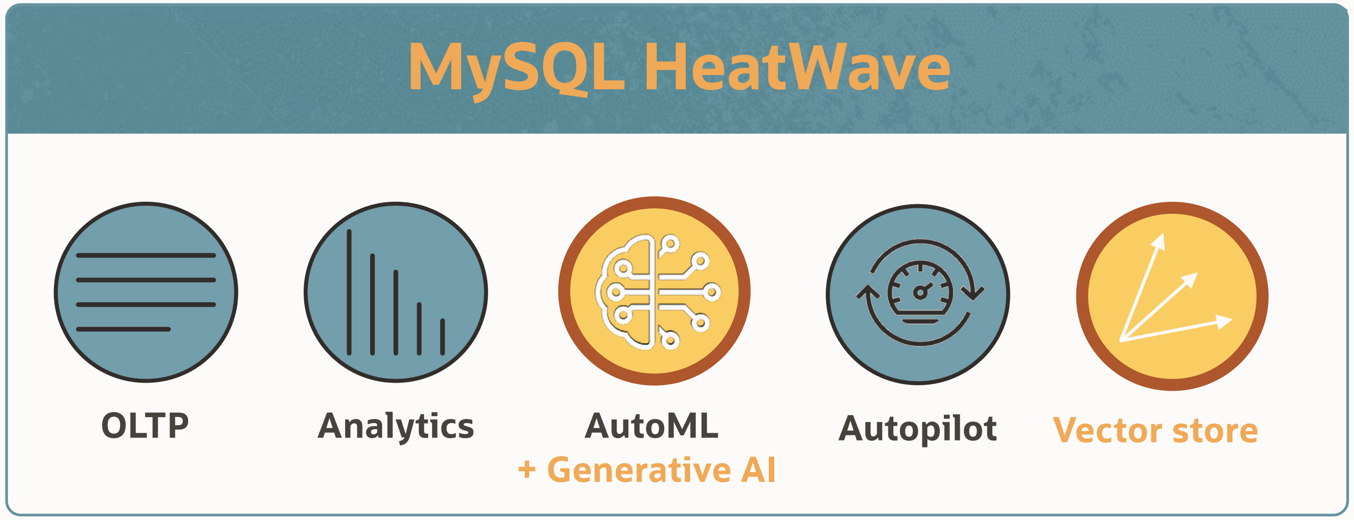
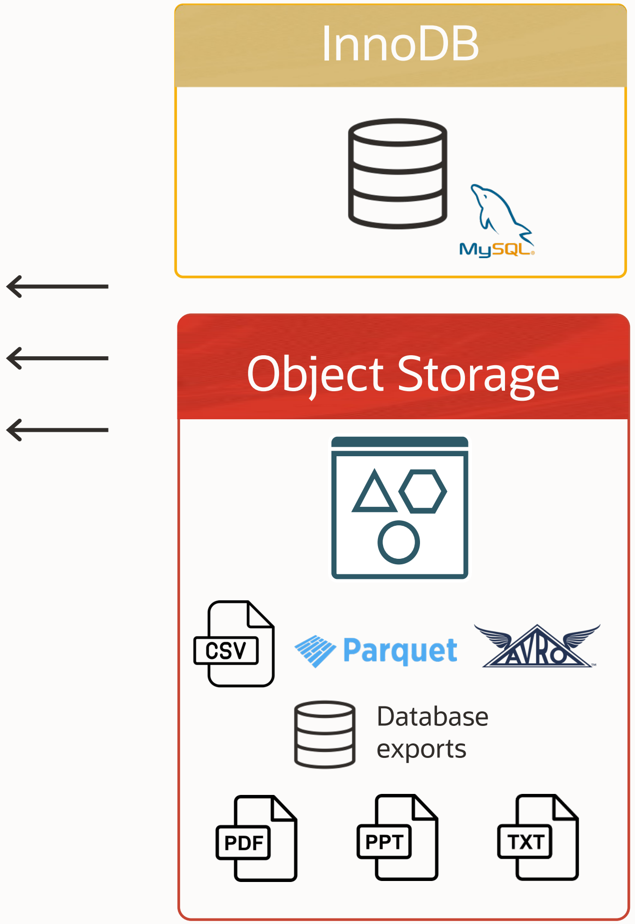
Developing GenAI and vector store applications with MySQL HeatWave

MySQL Belgian Days 2024

Matteo Casserini
Consulting Member of Technical Staff, Oracle

MySQL HeatWave Lakehouse





- Users can query and retrieve information in natural language
- Efficient searching of documents in object storage using vector store

Major Challenges in Generative AI

1) Large Language Models (LLMs) prone to **Hallucinations**

A plausible but false or misleading response generated by an AI algorithm

- ChatGPT *"an omniscient, eager-to-please intern who sometimes lies to you"**
- Some studies estimate chatbots to hallucinate **as much as 27%** of the time
- How to mitigate this inherent issue in LLMs?

* Prof. Ethan Mollick,
Wharton School of Business

Major Challenges in Generative AI

2) Incorporate Additional Information Sources in LLMs

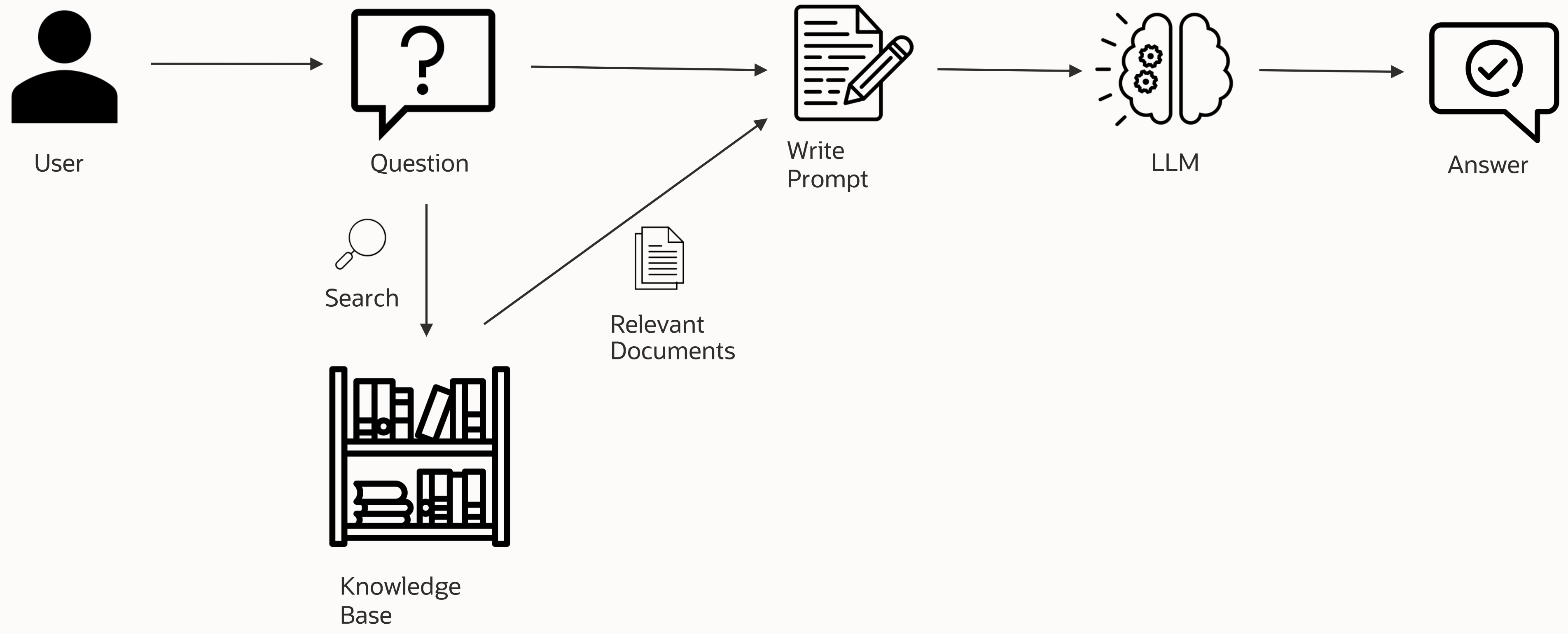
- At their core, LLMs can generate information only based on knowledge from their training data. 2 inherent limitations:
  - Given size and investment needed, training data tend to be out-of-date (ChatGPT: January 2022)
  - Pre-trained LLMs only trained on publicly available information (no business-specific info)
- In other words, LLMs generate answers only based on the *information memorized at training time* within the model and the *query* provided → 2 strategies to incorporate additional information
  - *Fine-tuning*: further train the LLM on additional training data (very costly, requires expertise)
  - *Grounding*: add additional relevant information as part of the query. Possible since LLMs have very large *context windows* (maximum number of tokens as input for text generation)

Meet Retrieval-Augmented Generation (RAG)

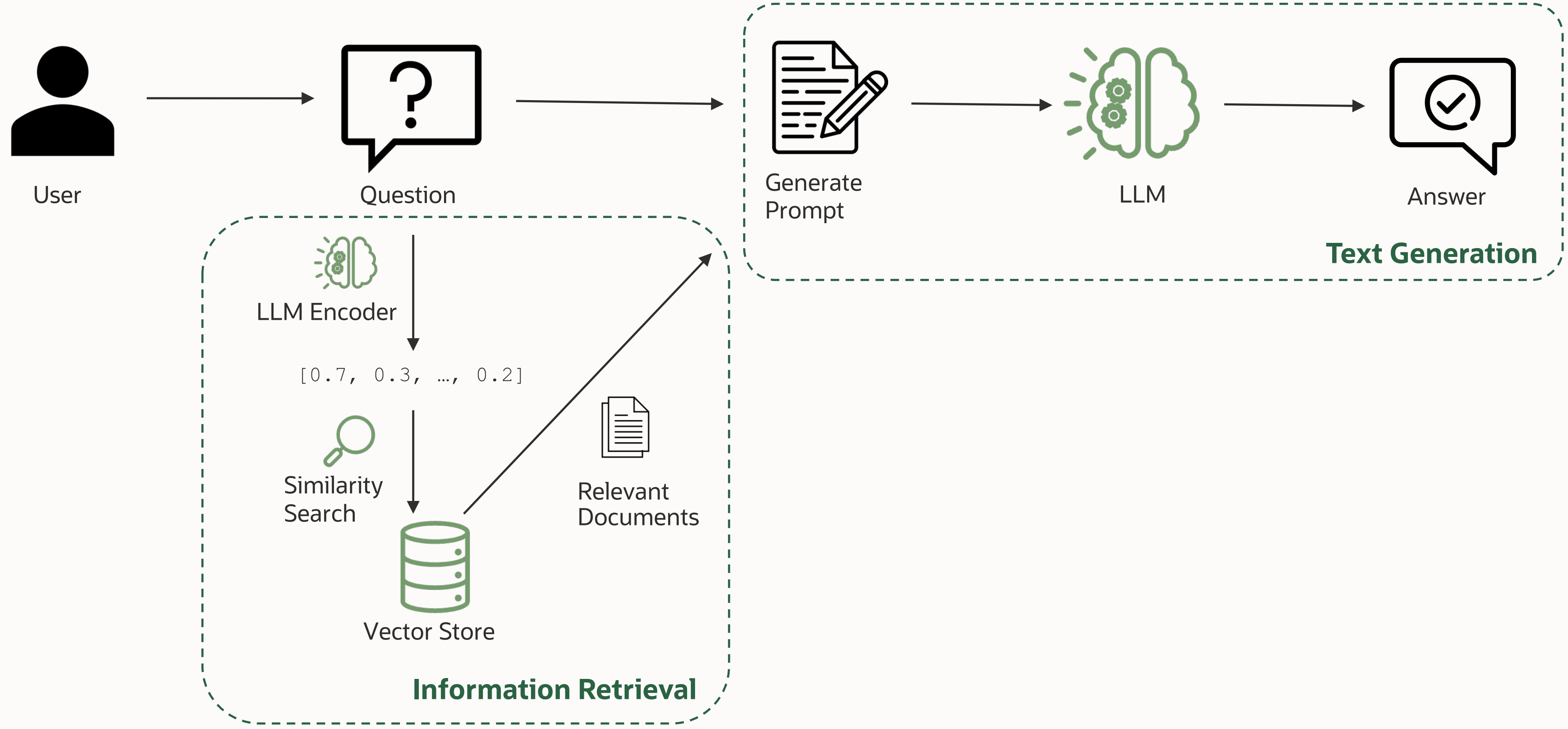
RAG is an LLM framework aiming to leverage the grounding process to solve both problems

- Generate **higher-quality** responses and **mitigate** hallucinations
  - Grounding also effective in reducing hallucination, especially when combined with prompt engineering
- **Automate** and make the grounding process **efficient**
  - How do we efficiently look for relevant information from external sources and incorporate it in the context window?

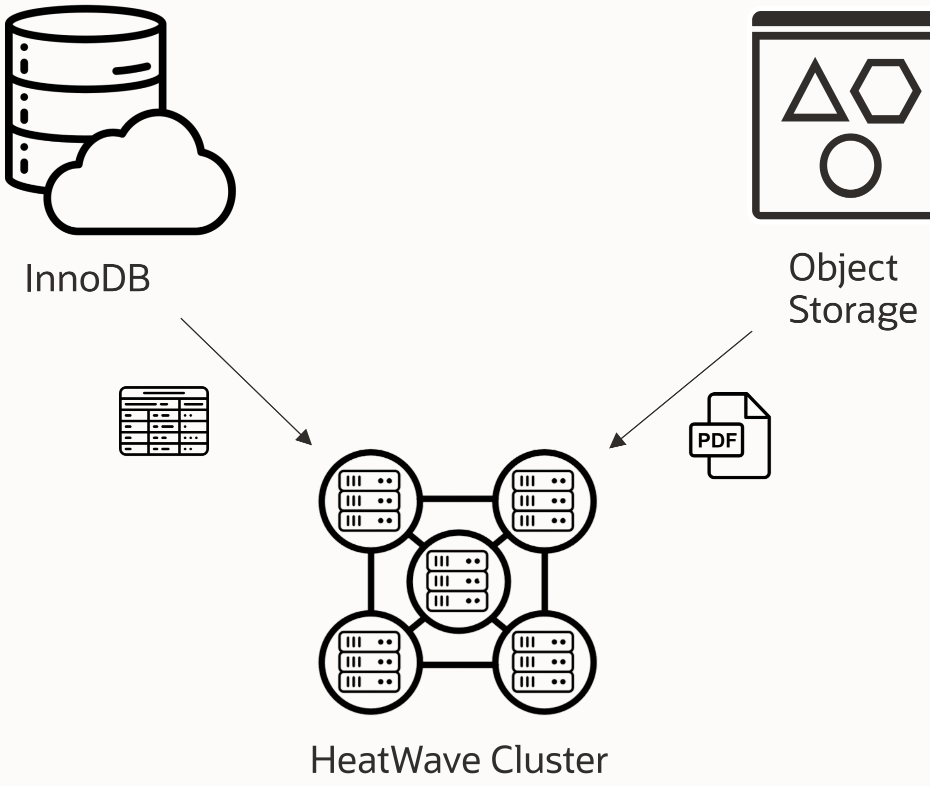
How Does “Manual” LLM Grounding Work?



How Does RAG Work?



Why MySQL HeatWave Lakehouse a Good Fit?



- Both OLAP and RAG aim to **answer user queries** based on **relevant information** from a **knowledge base**
- HeatWave right at the intersection of 2 important knowledge base types
  - Database tables
  - Unstructured documents in object storage

LLM Model Serving

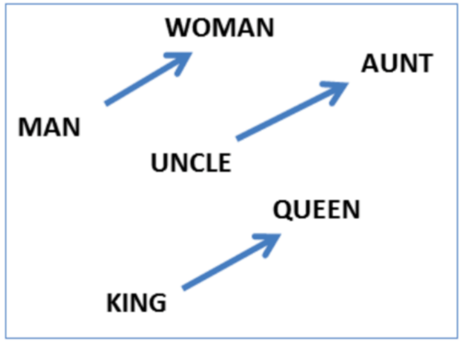
For RAG, we need to be able to **serve LLM models**

- MySQL HeatWave leverages the **OCI Generative AI service** (Beta, GA) with support for
  - Cohere LLM models (Command, Embed, Summarization)
  - Meta’s Llama2 model

Vector Store

Vector Store to manage **vector embeddings** from different knowledge bases

- Vector embeddings are generated by the LLM (encoder component)
  - Capture **semantics** of underlying text snippets



semantic: $v(\text{king}) - v(\text{man}) + v(\text{woman}) \approx v(\text{queen})$

- How to easily and efficiently populate vector store with such embeddings?
  - MySQL HeatWave: easy ingestion of documents in various formats (.pdf, .ppt, txt) from object storage

Image source: https://lena-voita.github.io/nlp_course/word_embeddings.html

Similarity Search

Vector embeddings capture semantics →

Most relevant documents for a user's query ≈ closest embeddings in the vector space

- Different ways to compute similarity of vectors: cosine distance, Euclidean distance…
- Computing similarities for all embeddings in a vector store can become costly
  - Various types of indices commonly used (e.g. IVF, HNSW…) for **approximate search** to improve performance

Thank you!

Q&A