MEM
GRAPH

Who's the most influential character in Game of Thrones?

The most influential character in Game of Thrones is Daenerys Targaryen.

✓ Cypher query - Successful:

```
MATCH (c:Character)-[r:KILLED]->(:Character)
RETURN c.name AS Character, sum(r.count) AS
Influence
ORDER BY Influence DESC
LIMIT 1
```

INTEGRATING KNOWLEDGE GRAPHS WITH GENAI:

How Memgraph Powers Real-Time Retrieval for LLMs

Table of Contents

Introduction

Everyone is talking about **Generative AI (GenAI)** and **Large Language Models (LLMs)**, and rightfully so. These technologies have changed the way we interact with machines, making them more intelligent, responsive, and human-like.

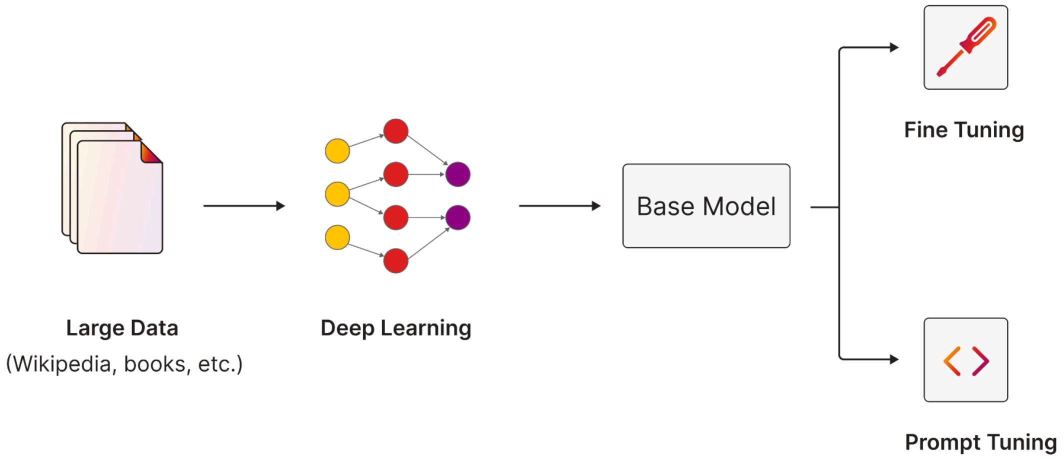
GenAI and LLMs have the power to generate content such as text, images, and videos based on simple natural language prompts. This capability is now transforming various industries, enabling applications from automated customer service to creative content generation.

However, the probabilistic nature of these models introduces significant risks, including **hallucinations**, **lack of transparency**, and reliance on **outdated information**. Given these challenges, how can we use the power of LLMs while mitigating these drawbacks? The answer lies in **augmenting these models with external data sources** and **sophisticated frameworks**—combining knowledge graphs, LLMs, and **Retrieval-Augmented Generation (RAG)**.

Before diving into the solutions, let's break down what LLMs are and why they are so impactful.

What is an LLM?

A **Large Language Model (LLM)** is a type of machine learning model designed to understand and generate human-like text. Trained on massive datasets, ranging from Wikipedia entries to entire books, LLMs can recognize patterns, predict word sequences, and generate coherent text. The broader the dataset and the more diverse the content, the better the model's performance.



LLMs use deep learning techniques to understand relationships between characters, words, and sentences, enabling tasks such as language translation, question-answering, and sentiment analysis.

Benefits of LLMs

LLMs are versatile, adapting to various applications, from creative content generation to customer service automation. They continuously improve with exposure to new data, learning in context to deliver better responses. Their adaptability makes them a powerful tool for rapidly scaling AI-driven solutions.

Limitations of LLMs

Despite their impressive capabilities, LLMs have several limitations that impact their effectiveness, especially in enterprise environments:

- Outdated information. LLMs provide insights based only on their training data, which may be outdated or incomplete.
- Training bias. LLMs reflect the biases inherent in their training data, potentially leading to skewed or unfair outcomes.
- Hallucinations. When encountering ambiguous or incomplete queries, LLMs may generate incorrect or fictitious information.
- Generic responses. Since most LLMs are trained on the same broad datasets, they provide generic answers. This limits their value in enterprise contexts, where custom, domain-specific insights are critical. Your LLM is essentially the same as anyone else’s, which restricts its adaptability for specialized tasks.

Addressing the Limitations: The Role of RAG

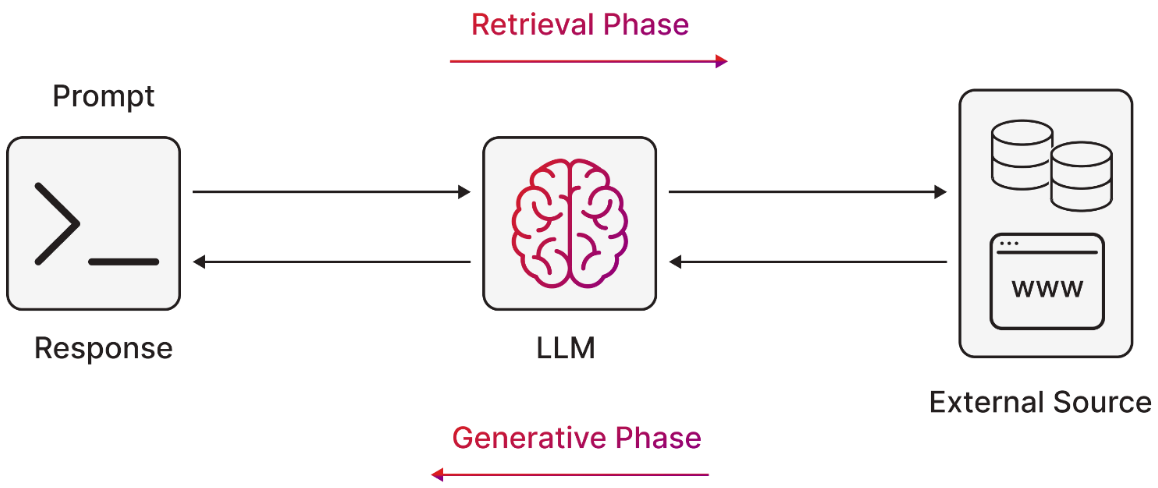
Retrieval-Augmented Generation (RAG) helps address these limitations by incorporating external data sources into LLM workflows. While RAG enhances the factual grounding and relevance of responses by utilizing real-time, verifiable data, it does not guarantee complete accuracy.

How Does RAG Work?

RAG operates through a two-phase process—retrieval and generation.

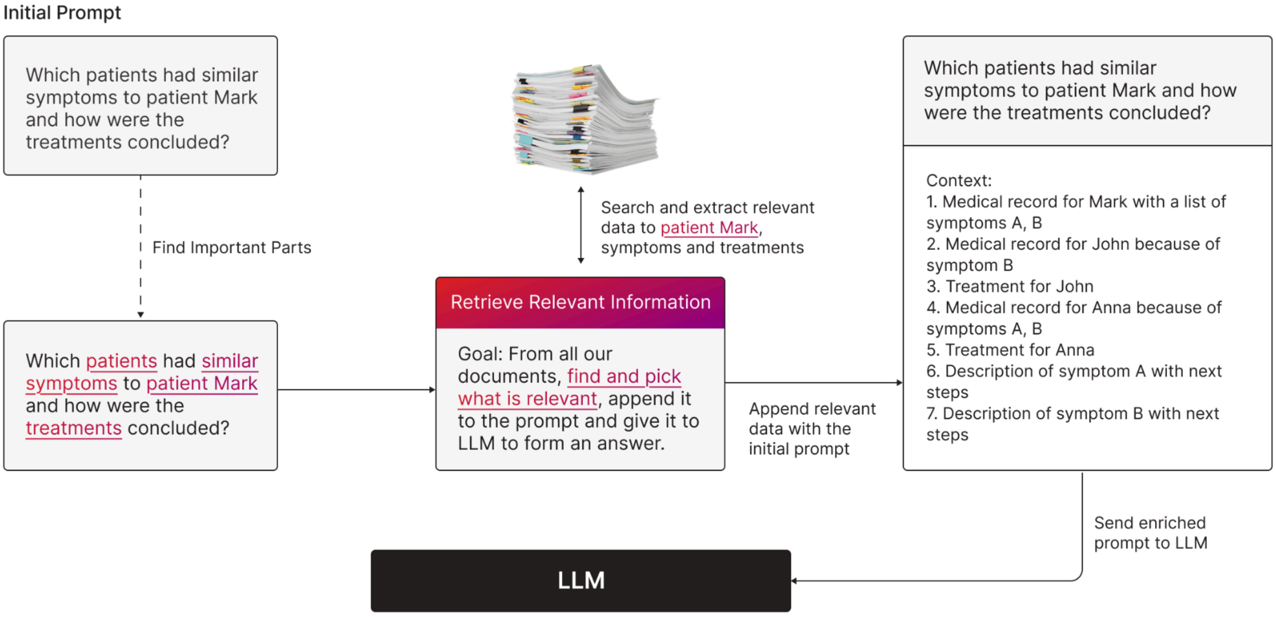
Retrieval Phase

The system processes the user’s query, retrieves relevant data from external databases or proprietary sources, and selects the most contextually appropriate information.



Generative Phase

The LLM uses the retrieved data to create an augmented response that is more accurate and grounded in real-world facts.


Initial Prompt
Which patients had similar symptoms to patient Mark and how were the treatments concluded?
Find Important Parts
Which patients had similar symptoms to patient Mark and how were the treatments concluded?
Search and extract relevant data to patient Mark, symptoms and treatments
Retrieve Relevant Information
Goal: From all our documents, find and pick what is relevant, append it to the prompt and give it to LLM to form an answer.
Append relevant data with the initial prompt
Which patients had similar symptoms to patient Mark and how were the treatments concluded?
Context:
1. Medical record for Mark with a list of symptoms A, B
2. Medical record for John because of symptom B
3. Treatment for John
4. Medical record for Anna because of symptoms A, B
5. Treatment for Anna
6. Description of symptom A with next steps
7. Description of symptom B with next steps
Send enriched prompt to LLM
LLM

Applications of RAG

Information retrieval. Improves the accuracy and relevance of search results.

Question-answering systems. Enhances response precision in specialized fields, such as healthcare and law.

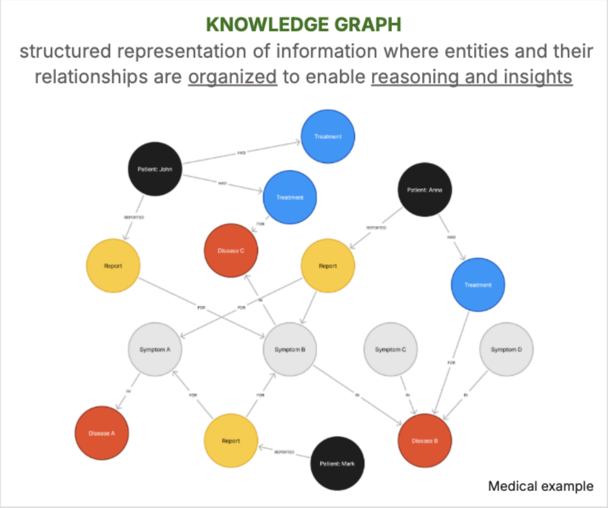
Conversational agents and chatbots. Provide more reliable, context-driven chatbot interactions.

Introducing GraphRAG: A Step Further with Knowledge Graphs

While **RAG** enhances LLMs, integrating **knowledge graphs** takes it to the next level. By organizing data as nodes and relationships, **knowledge graphs** offer a structured framework, making it easier for LLMs to deliver contextually rich, accurate responses.

What is a Knowledge Graph?

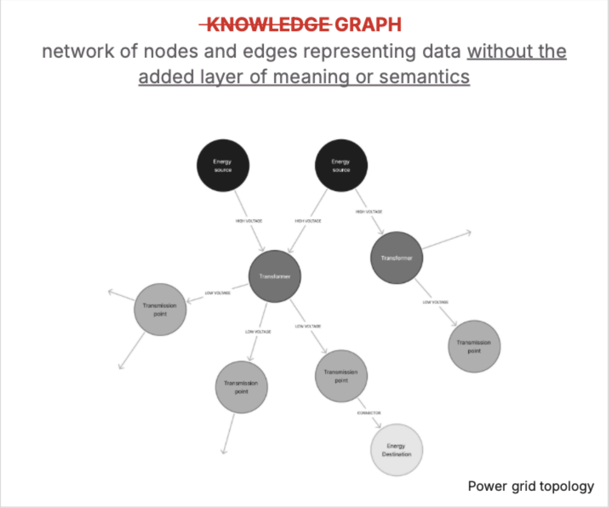
A **knowledge graph** is a data structure that captures relationships between different entities. These graphs provide LLMs with structured, real-time access to **contextual data**, reducing the likelihood of hallucinations and improving response accuracy.



KNOWLEDGE GRAPH

structured representation of information where entities and their relationships are organized to enable reasoning and insights

Medical example



~~KNOWLEDGE~~ GRAPH

network of nodes and edges representing data without the added layer of meaning or semantics

Power grid topology

How GraphRAG Works?

GraphRAG combines **knowledge graphs** with **RAG** for smarter data retrieval and response generation:

- **Retrieval phase**. The system uses the **knowledge graph** to traverse relevant entities and relationships.
- **Generative phase**. The **LLM** uses the enriched context from the knowledge graph to generate more accurate and insightful responses.

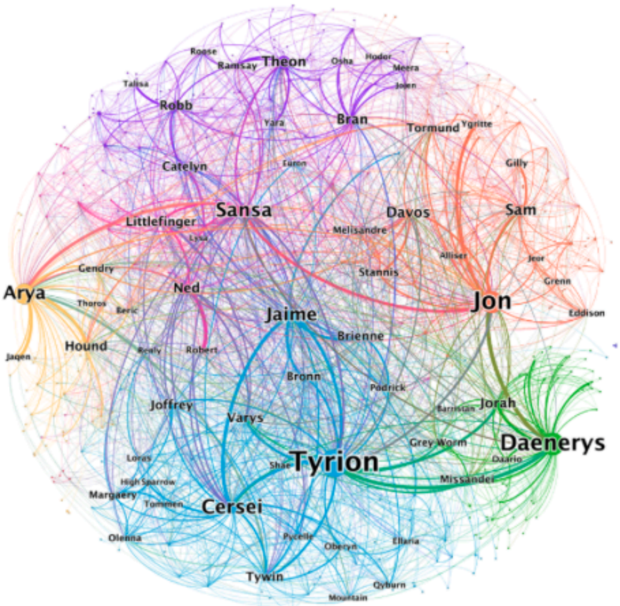
Building GraphRAG apps with Memgraph?

Memgraph is the foundation for building GraphRAG systems, thanks to its high-performance, in-memory graph database architecture. With Memgraph, developers can efficiently store and query knowledge graphs in real-time, ensuring that the data fed into LLMs is both current and contextually relevant.

Why combine knowledge graphs, RAG, and LLMs?

GraphRAG is the ideal synergy between LLMs and knowledge graphs. It addresses the key limitations of traditional LLMs by enabling accurate, real-time responses grounded in structured data. Memgraph amplifies this process, offering the tools to handle complex queries and dynamic data.

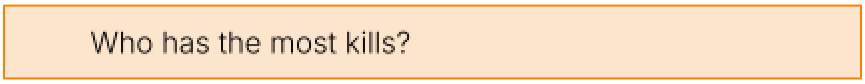
For example, in a Game of Thrones query asking "Who has the most kills?", a basic LLM might guess Arya Stark based on pattern recognition. However, when augmented with a knowledge graph containing kill counts, the system can accurately return Daenerys Targaryen with 1,044 kills.



https://studentwork.prattsi.org/infovis/visualization/network-graphs-of-game-of-thrones-the-tv-series/

Let’s take our Game of Thrones made in our Memgraph Playground as an example. Here you have a query: "Who has the most kills?"

PROMPT



Who has the most kills?

CYPHER

```
MATCH (c:Character)-[r:KILLED]->(:Character)
RETURN c.name AS Character, SUM(r.count) AS Kills
ORDER BY Kills DESC
LIMIT 1;
```

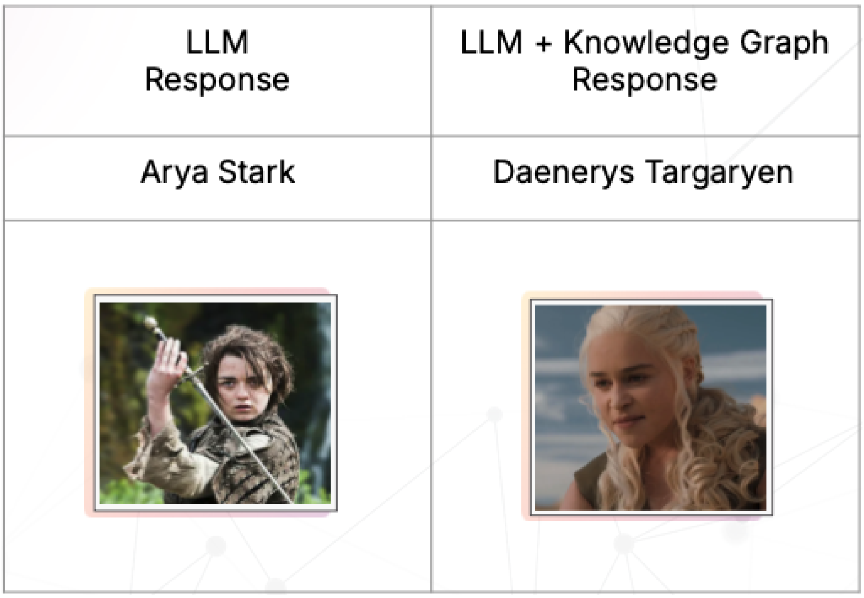
An LLM alone might incorrectly answer 'Arya Stark' when asked which character in *Game of Thrones* has the most kills, due to its reliance on general knowledge and pattern recognition. Arya Stark is known for being a skilled assassin, which could lead the model to select her based on its training data.

However, when the LLM is augmented with a knowledge graph that includes detailed information, such as the number of kills attributed to each character, it would correctly identify Daenerys Targaryen with 1,044 kills.

This example illustrates how knowledge graphs can enhance the accuracy and contextual relevance of LLM responses by providing access to specific, structured data that goes beyond the general patterns found in text.

RESULT

```
Character: "Daenerys Targaryen"; Kills: 1044
```



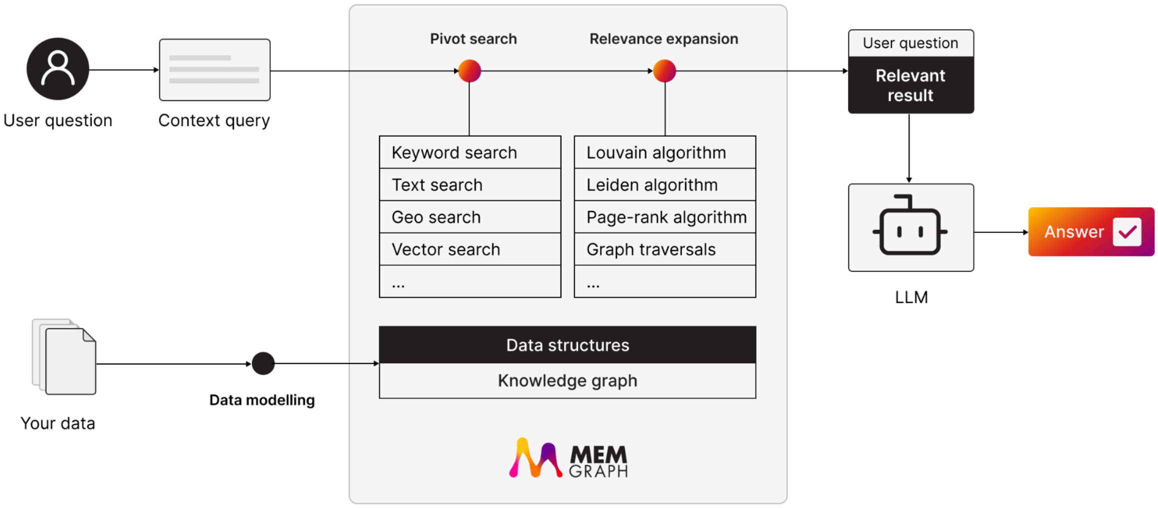
LLM Response	LLM + Knowledge Graph Response
Arya Stark	Daenerys Targaryen

How to Build GraphRAG with Memgraph

Building GraphRAG systems with Memgraph involves:

- Efficient data modeling. Use Memgraph to structure data into knowledge graphs.
- Algorithmic power. Leiden, PageRank, and graph traversal algorithms ensure that the most relevant data is retrieved and processed efficiently.
- Integrations. Use LangChain and LlamaIndex to streamline RAG workflows by connecting Memgraph's graph database with LLMs.

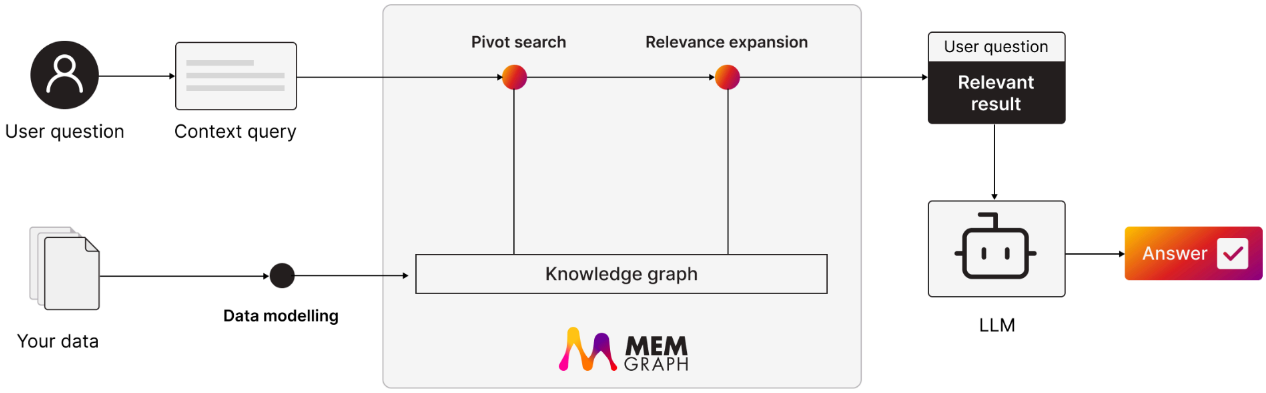
For more details, visit our Memgraph docs: Memgraph AI Ecosystem.



The Role of Memgraph in GraphRAG

Memgraph's real-time graph analytics and property graph model offer flexibility in data representation. Memgraph's in-memory architecture ensures real-time performance, making it perfect for dynamic, large-scale knowledge graphs.

Memgraph's integration with LangChain JS, LangChain Python, and LlamaIndex simplifies the process of building RAG systems, allowing developers to focus on creating responsive AI applications.



Why Memgraph Enterprise?

Memgraph's **Enterprise Edition** offers advanced features for developers looking to scale their **GraphRAG** systems:

- **Dynamic algorithms**. These algorithms allow real-time updates to the graph without reprocessing the entire dataset, ensuring fast response times as new data streams in.

- **Time-to-Live (TTL)**. Automatically expire nodes and relationships to keep your knowledge graph lean and relevant.

- **L3 support**. Direct access to engineers, not just support tiers, ensuring faster issue resolution.

- **Influence the roadmap**. Collaborate with Memgraph's team to tailor features to your needs.

- **CTO on speed dial**. Get real-time access to Memgraph leadership for direct support and project guidance.

Conclusion

Memgraph's GraphRAG empowers developers to build smarter, more reliable **GenAI** apps. By combining **LLMs** with **knowledge graphs**, Memgraph eliminates the common limitations of traditional AI, offering accurate, context-aware responses based on **real-time, structured data**.

GraphRAG represents the future of AI—one where LLMs and knowledge graphs work together to provide deeper insights, fewer hallucinations, and more actionable answers.

Next steps?

Read how companies like Microchip, Precina Health, and Cedars-Sinai use Memgraph to build GraphRAG genAI apps in our latest user stories:

- Enhancing LLM Chatbot Efficiency with GraphRAG (GenAI/LLMs)
- How Precina Health Uses Memgraph and GraphRAG to Revolutionize Type 2 Diabetes Care with Real-Time Insights
- Cedars-Sinai: Using Graph Databases for Knowledge-Aware Automated Machine Learning
- Memgraph Lab 101: Simplify Graph Data Exploration with Visualization and Querying
- Building GenAI Applications with Memgraph: Easy Integration with GPT and Llama

Memgraph Academy

If you are new to the GraphRAG scene, check out a few short and easy-to-follow lessons from our subject matter experts. For free. Start with:

- Introduction to Large Language Models
- Understanding RAG
- The Role of Knowledge Graphs