Onepoint TechTalk

# Decoding AI series

Session 5

## The future of enterprise data access- RAG, GraphRAG and beyond

Our awesome speaker

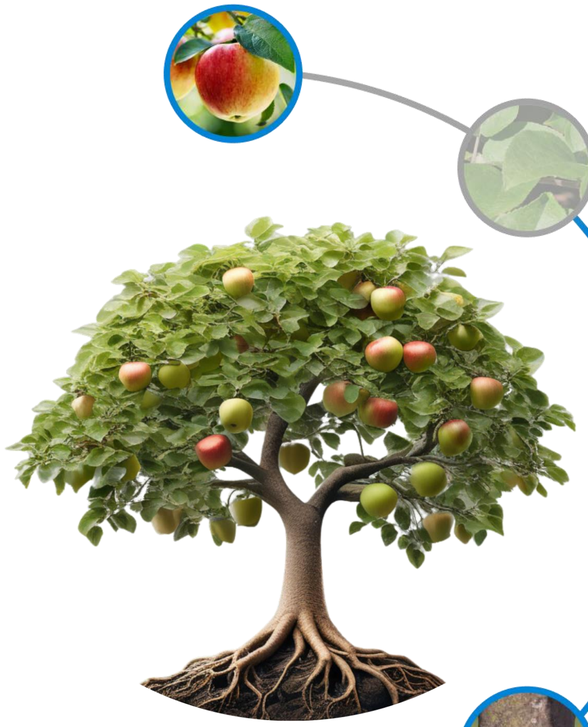
Gil Fernandes

AI Solutions Engineer

Welcome

# From Roots to Fruits

- Journey to create value from LLMs



**Fruit**

3. How to turn the spotlight on Dark Data with Gen AI

**Leaves**

6. AI agents
7. Ensemble of agents
8. Building complex simulations
9. Building LLMs based systems

**Branches**

4. Workflows and complex interactions

5. RAG, GraphRAG and beyond

**Trunk**

2. Direct interactions with LLMs

**Roots**

1. Fundamentals of LLMs

# Agenda

# Insights from previous webinar

# Audience poll

How are you accessing your documents?

A. In paper format
B. In electronic format (e.g. file systems, Google Docs)
C. We use a documentation portal (e.g. Confluence, Sharepoint)
D. We use an AI based tool to chat with our documents
D. Enterprise Content Management Systems (ECMS)
E. Document Databases (NoSQL, line MongoDB or Graph Databases)
F. Others

Insights from previous webinar

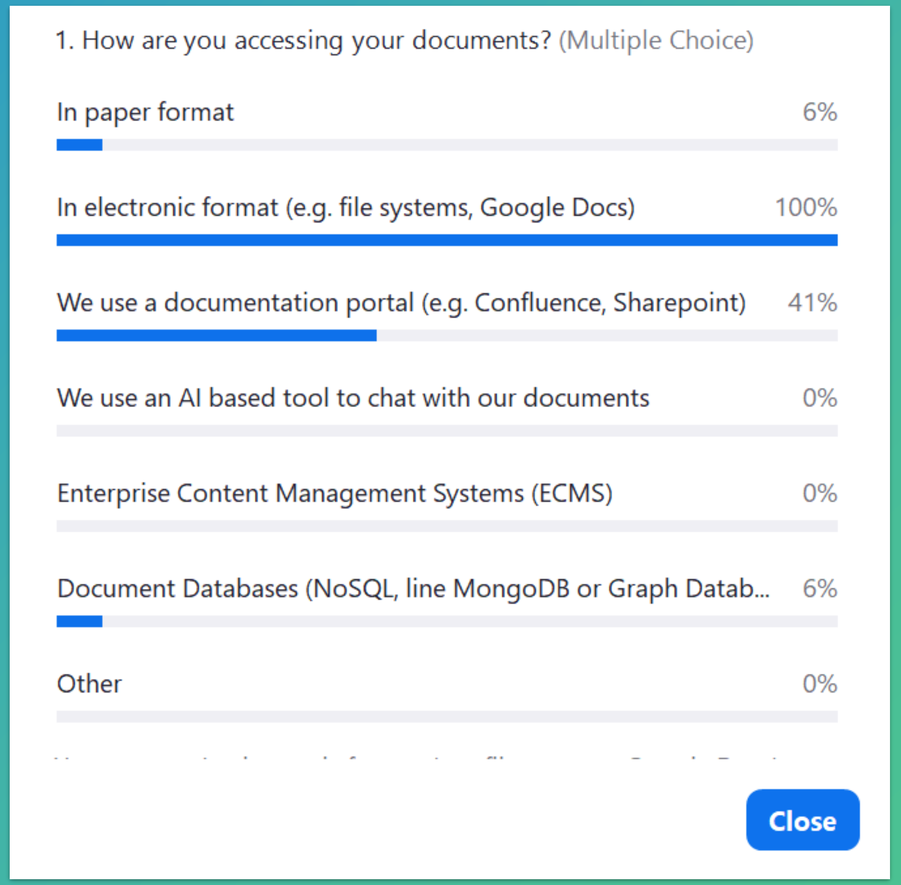
# LLMs suited for many scenarios

- Great at a variety of NLP tasks
- Converting unstructured data into structured data
- Tackling the problem of “dark data”
- Learning from context

- Limited reasoning
- Knowledge cut-offs
- Long-term planning
- Missing sources
- Hallucinations
- Refusals

https://www.onepointltd.com/techtalk

# Audience insights



# What is RAG?

# RAG paper

The concept of RAG was introduced in 2021 by this paper:

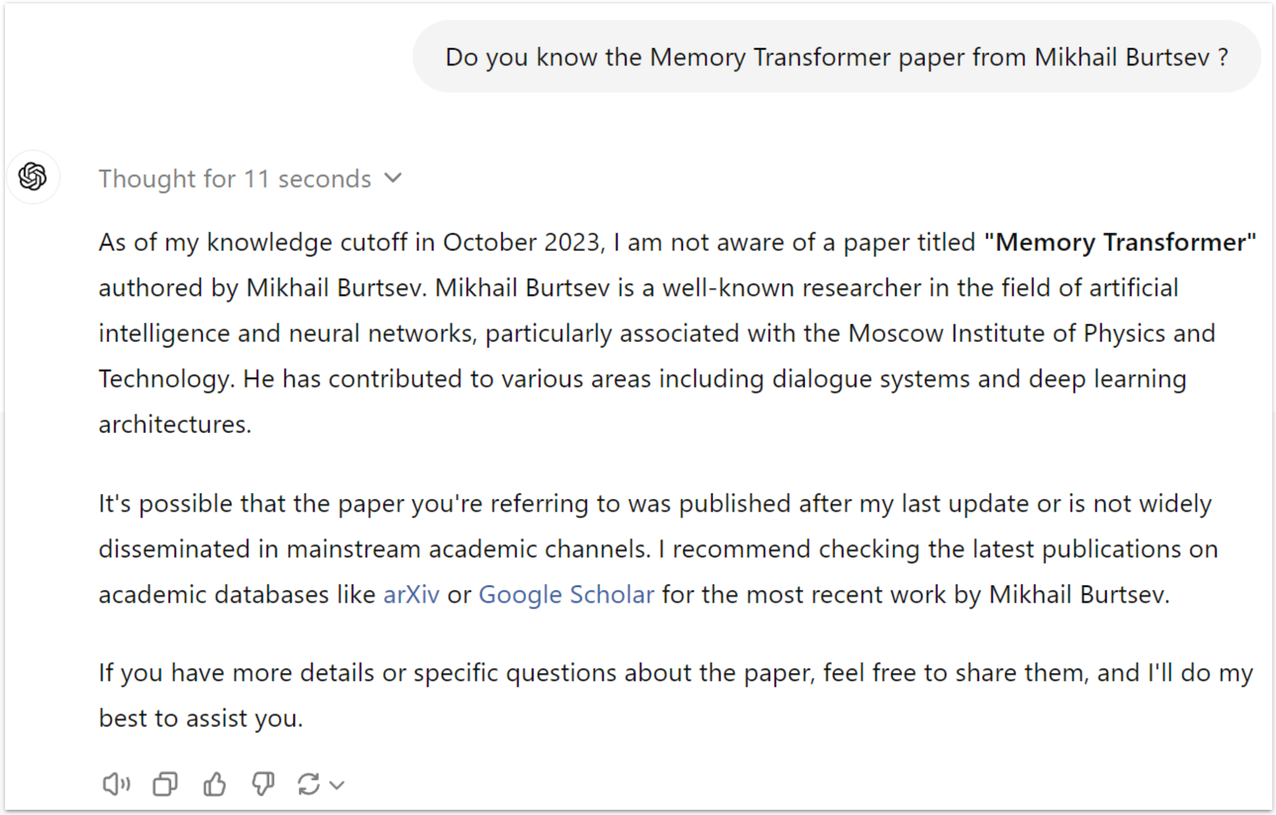
“Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks”

This paper states that

> *“For language generation tasks, we find that RAG models generate more specific, diverse and factual language than a state-of-the-art parametric-only seq2seq baseline.”*

# RAG why?

LLMs lack recent or specialised knowledge and cannot answer queries.




Source: ChatGPT o1-preview

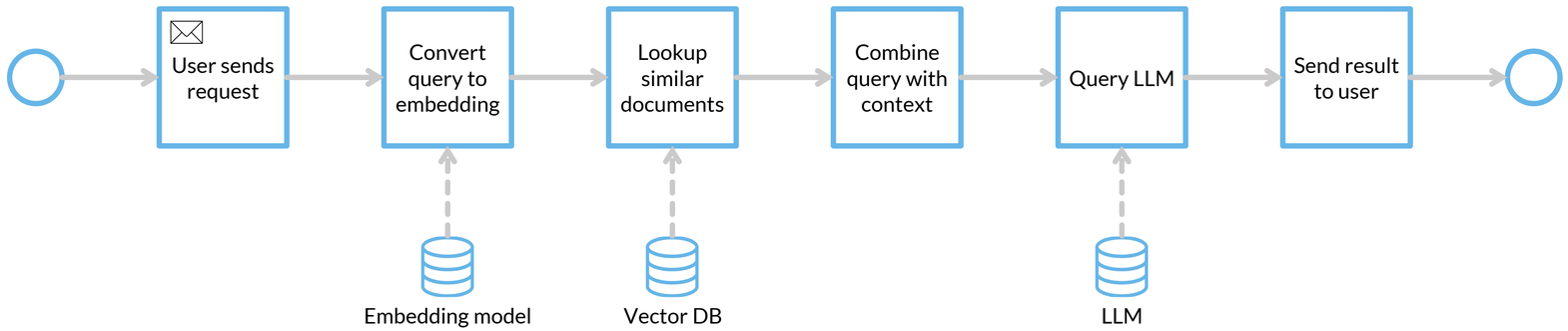
RAG (Basic)

# Retrieval Augmented Generation

Enhance LLM knowledge with
private data and vector databases

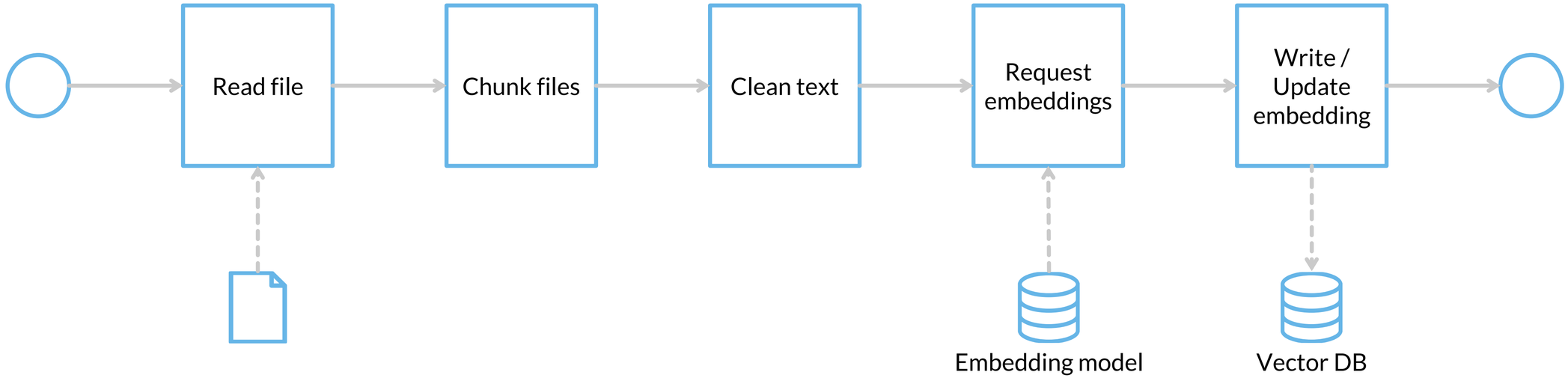
## Involved components

- LLM
- Vector database
- Orchestration



# Retrieval Augmented Generation

RAG data needs to be “indexed”, i.e. converted to vectors which have to be generated by the LLM.



# How to use RAG in your enterprise?

# How to use RAG in your enterprise?

- Talk to your documents
- Search augmentation
- Customer self-service
- Product search
- HR and Talent management
- Personalised marketing

RAG

# Talk to your documents

## Problem

You spend extensive time reviewing documents and drafting contracts.

## Solution

RAG systems can document databases or contract templates for precedents, clauses, and terms and then generate drafts or provide insights into compliance and risks. This could be interesting for legal firms.

RAG

# Search augmentation

## Problem

You want to search using natural language using summarisation

## Example


## Solution

RAG systems allow to understand normal questions, not just keywords and transform search results by using NLP features, like e.g., summarisation, translation, etc.

# Customer self-service

## Problem

You have a large Wiki for customer support and want to make it more easily searchable.

## Solution

RAG systems allow to ask questions in natural language and to access information in the Wiki in a much more natural way, like when interacting with a well-informed agent.

RAG

# Product search

## Problem

You have a large retail database and want to allow customers to search it using natural language and also to get well informed answers with suggestions.

## Solution

RAG systems allow to ask questions in natural language and to access information in the Wiki in a much more natural way, like when interacting with a well-informed agent.

# HR and Talent management

## Problem

HR teams struggle to match job requirements with the right candidates quickly and accurately.

## Solution

RAG systems can retrieve candidate information (e.g., resumes, past performance) and relevant industry standards to generate optimised job descriptions or match candidates with roles.

# Personalised marketing

## Problem

Generic marketing content lacks engagement and may not align with user preferences.

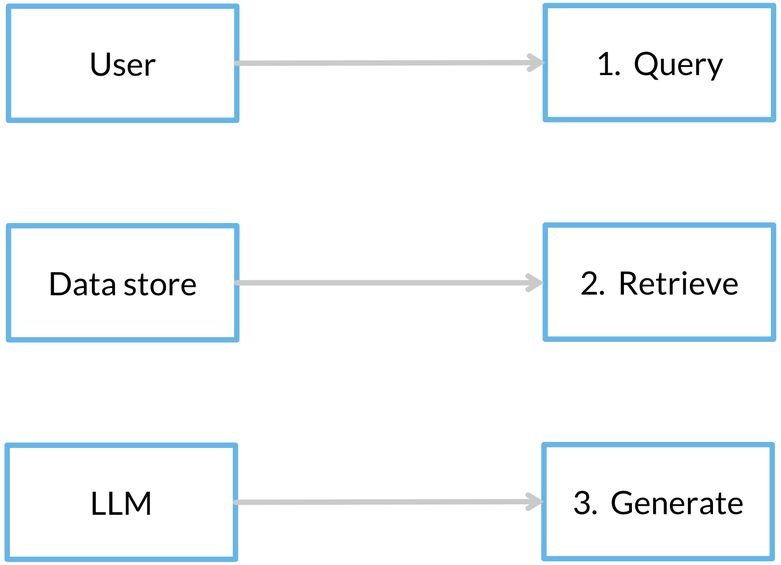
## Solution

RAG systems can retrieve candidate information (e.g., resumes, past performance) and relevant industry standards to generate optimised job descriptions or match candidates with roles.

# Enhancing RAG

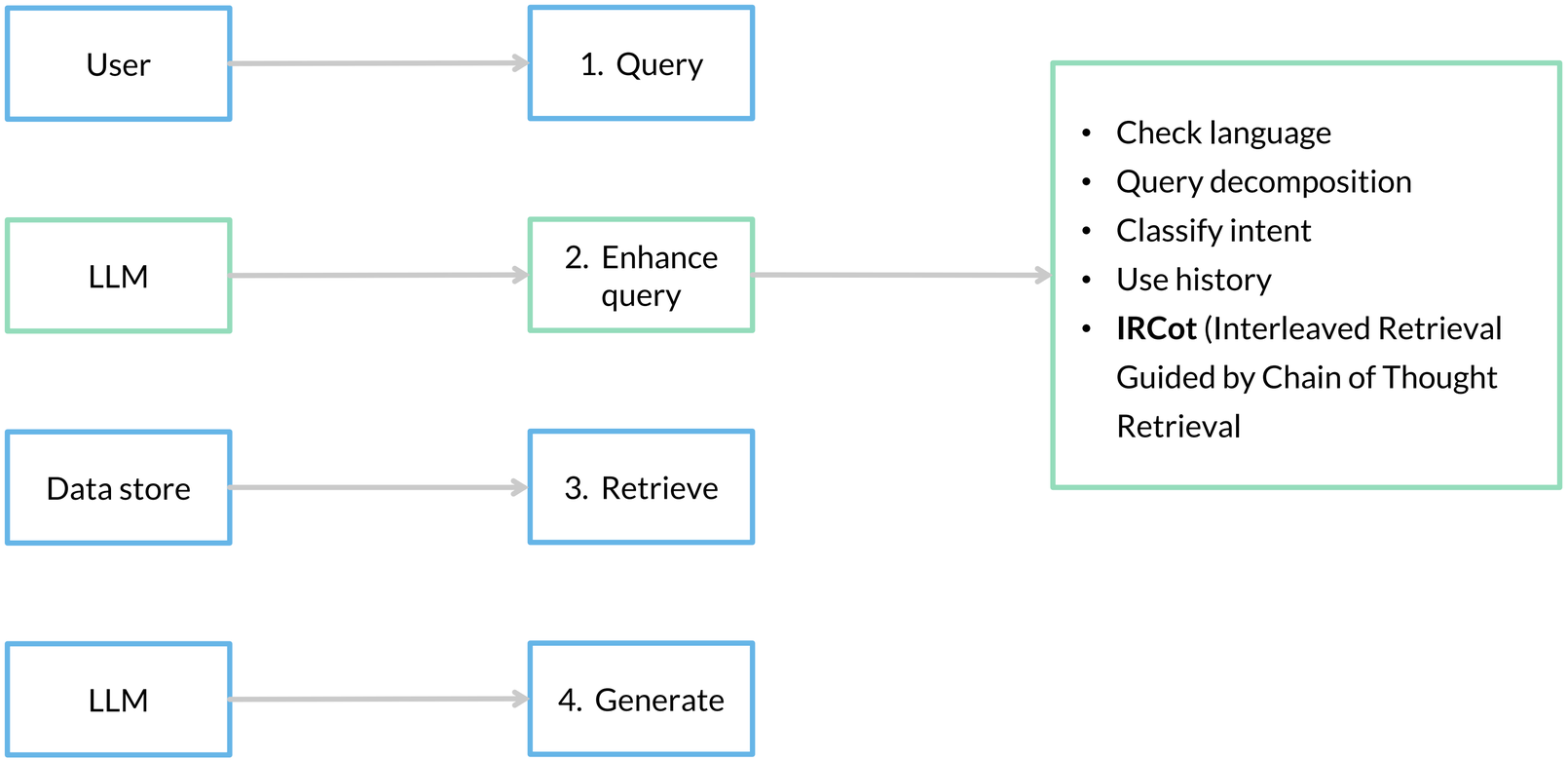
# Enhancing RAG - Retrieval (1 of 5)

Basic RAG retrieval has these basic steps:



# Enhancing RAG - Retrieval (2 of 5)

We add additional query enhancement:



# Enhance query example

- You have multiple tasks based on a query. The first is about detecting the **intent** of the user's query.
- You need to return the intent and the confidence of the intent detection model.
- You also extract **keywords** from the user's query.
- You need to return the keywords extracted from the user's query.
- Finally, you also break down the user's query into **subqueries**.
- You need to return the subqueries that can the user query can be broken into

For example:

**Query**: "What is the weather in Tokyo?"

**Intent**: weather, confidence: 0.9

**Keywords**: Tokyo, weather

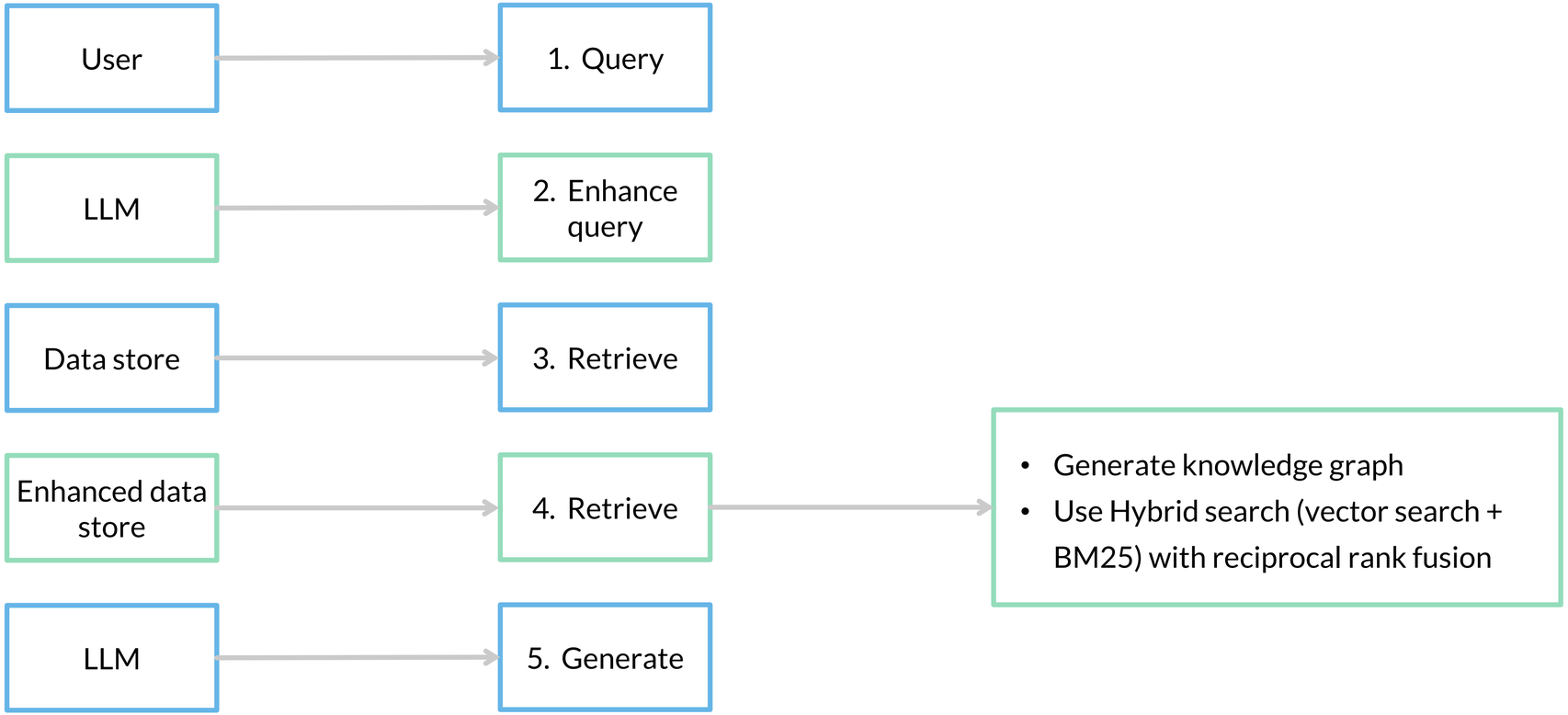
**Subqueries**: What is the temperature in Tokyo?, What is the humidity in Tokyo?, Will it rain in Tokyo?

Here is the query:

{query}

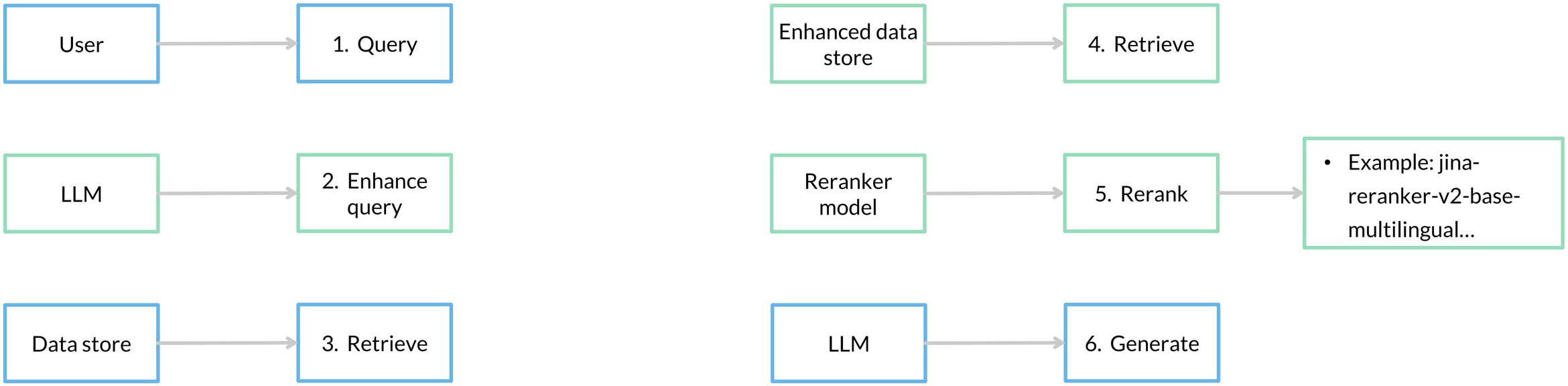
# Enhancing RAG - Retrieval (3 of 5)

We can add a secondary data store which is in many cases a knowledge graph:



# Enhancing RAG - Retrieval (4 of 5)

We can use a reranker model to filter out parts of the context which are not so relevant:



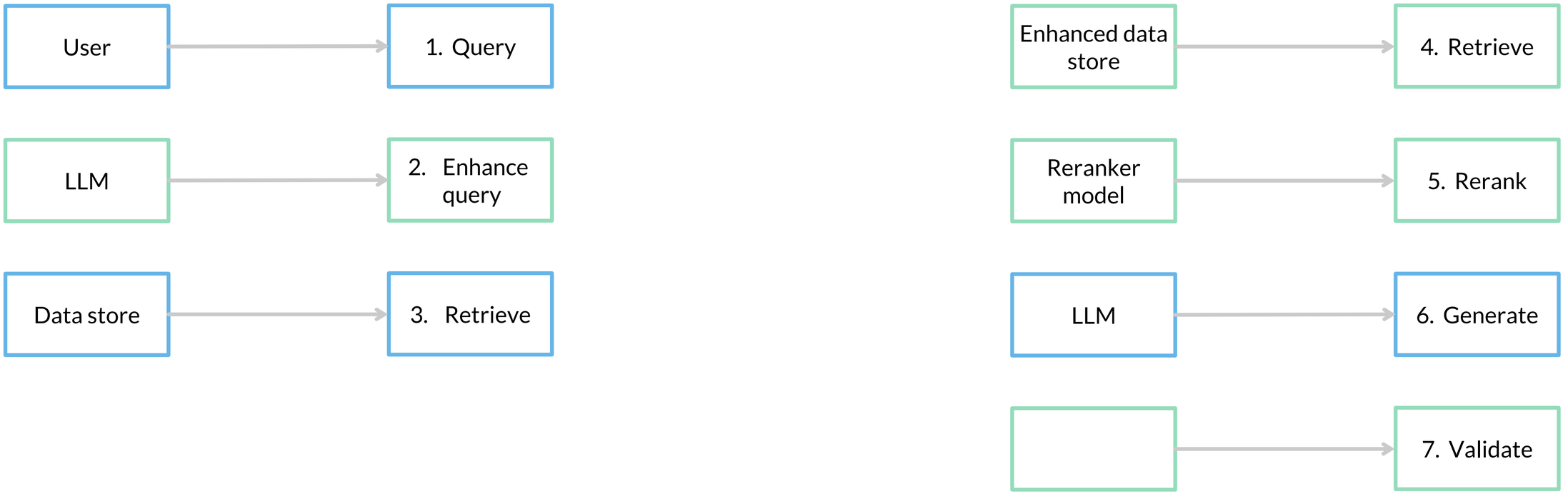
Enhancing RAG

# jina-reranker-v2-base-multilingual

## Re-ranker models can be used via API's. Example:

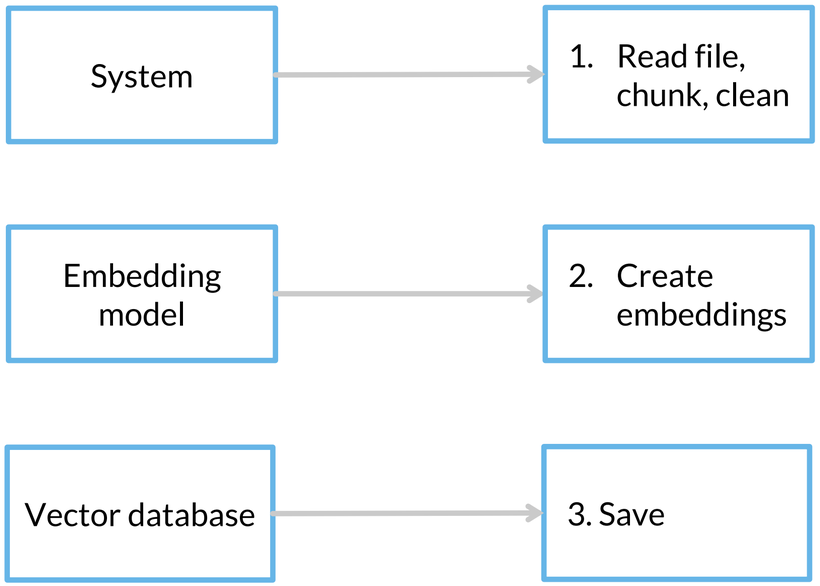
# Enhancing RAG - Retrieval (5 of 5)

A validation step can be added at the end of this sequence:



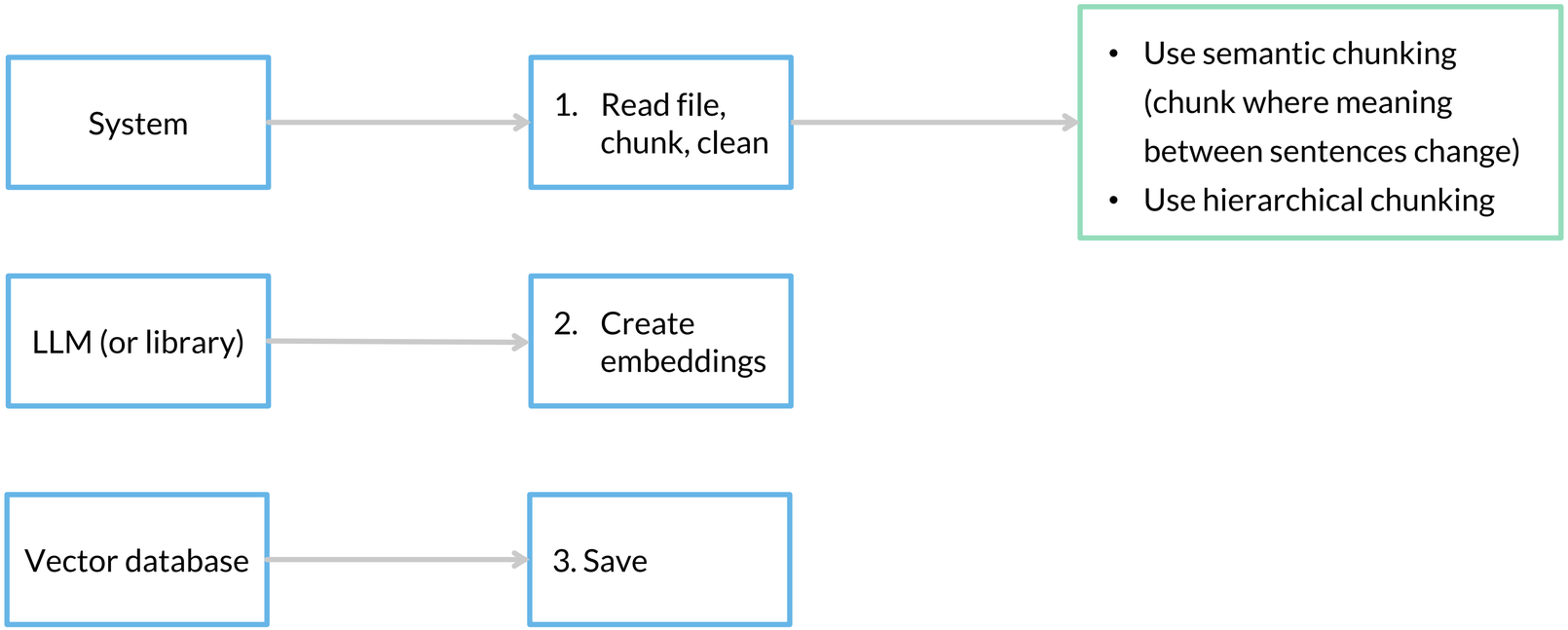
# Enhancing RAG - Indexing (1 of 5)

Basic RAG has these basic steps:



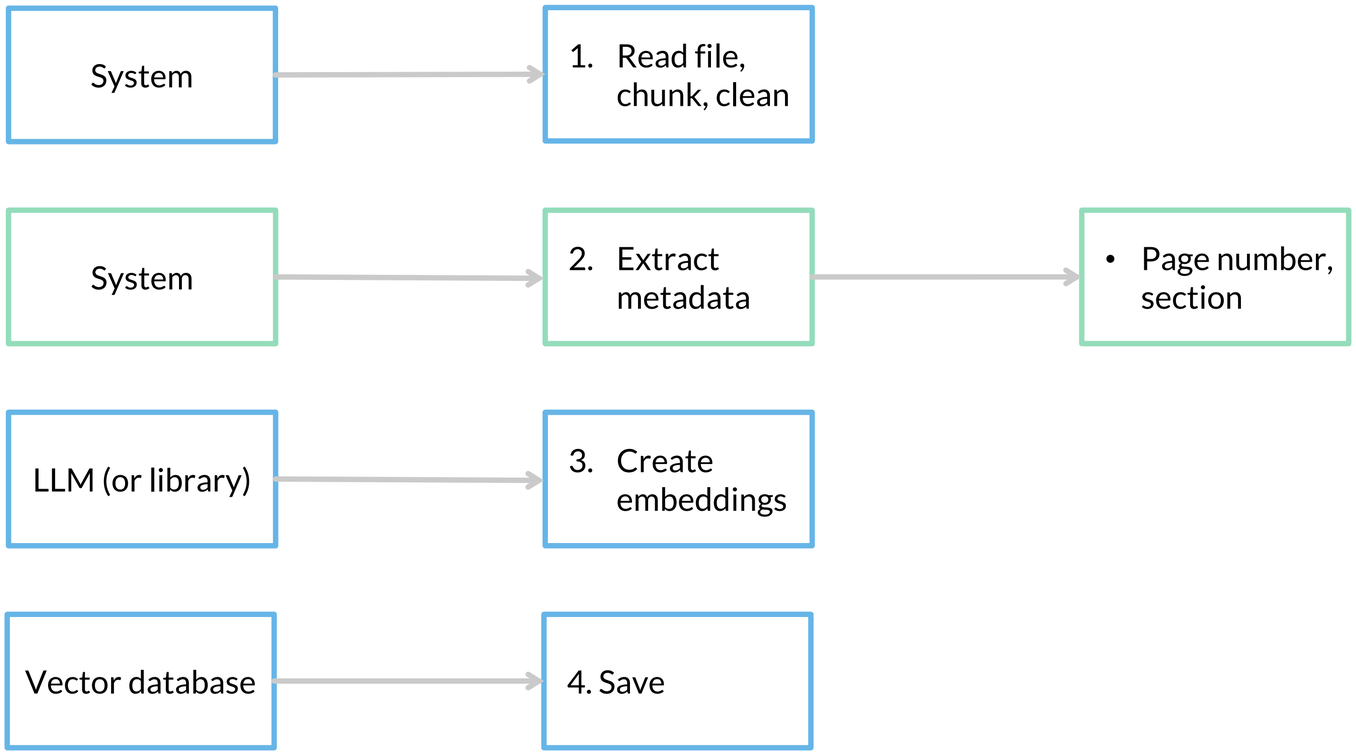
# Enhancing RAG - Indexing (2 of 5)

Basic RAG has these basic steps:



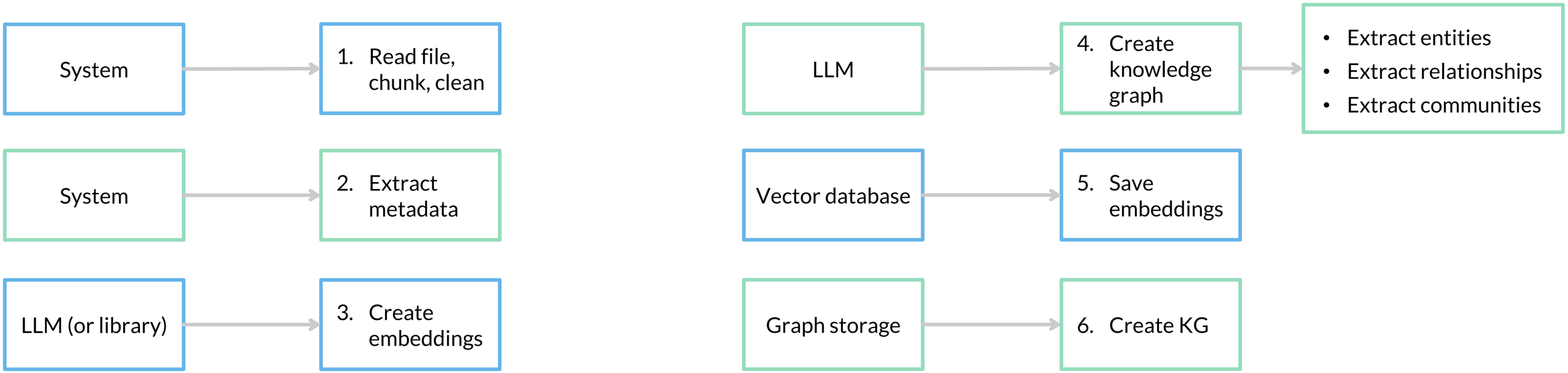
# Enhancing RAG - Indexing (3 of 5)

Basic RAG has these basic steps:



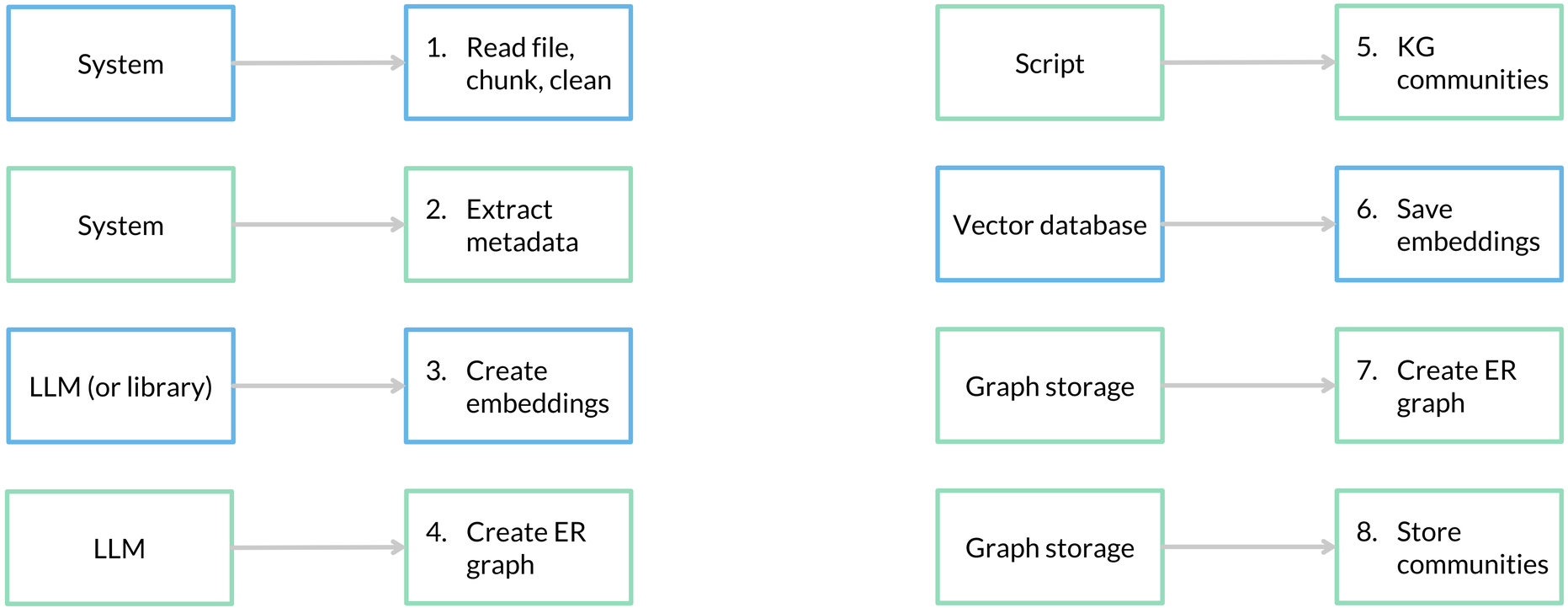
# Enhancing RAG - Indexing (4 of 5)

You can also generate a knowledge graph to represent the relationship between entities:



# Enhancing RAG - Indexing (5 of 5)

Knowledge graphs allow to extract communities with summaries. You can also store these.



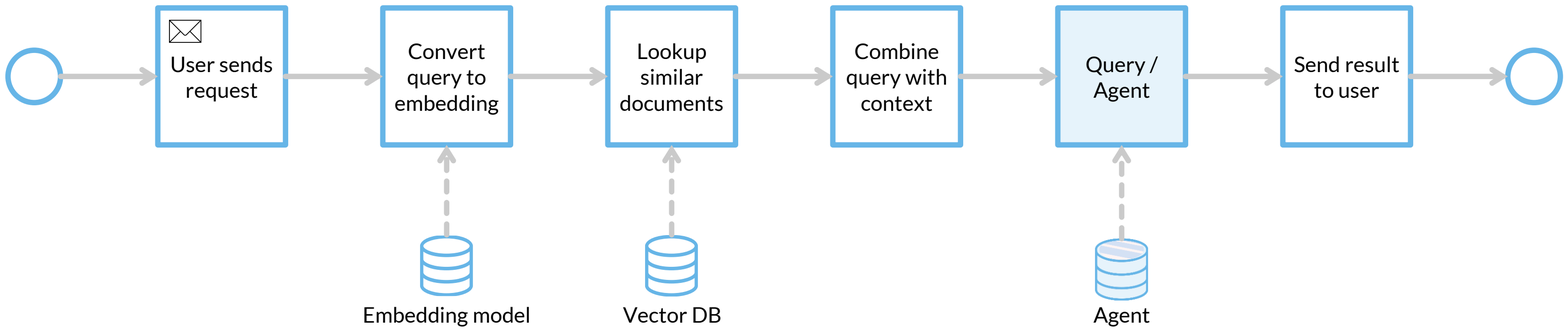
# Agentic RAG

# Agentic RAG

- LLMs have a lot of knowledge, but they have knowledge cut offs and have no access to real time data.
- LLM based agents can however overcome these limitation and access real time data.
- So we just **replace the LLM with an agent**.

## Involved components

- Agent (LLM + tools)
- Vector database
- Orchestration



# Evaluating RAG

# What to evaluate?

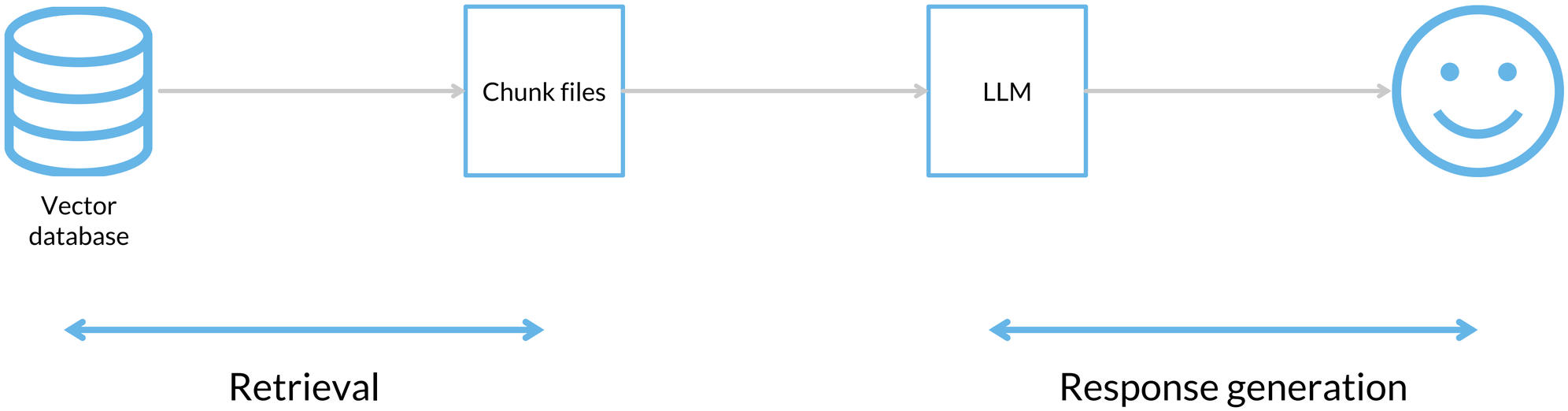
## Retriever

This is the component you use for searching, like the vector database.

## Response generator

The result of the RAG system is being evaluated in this case.

# What to evaluate?



# Retriever evaluation

## ML evaluation

Classic machine learning metrics used:

- Hit rate, MRR, Precision, Recall, F1
- Requires for a ground truth, i.e. the creation of a dataset to be available

## LLM evaluation

- Uses the LLM to score documents based on their relevance and ranking to a given question.
- Does not require the creation of a dataset.

# Retriever evaluation example

## ML evaluation

In this case you retrieve two out of 3 documents correctly and you get different scores for different metrics:

Hit rate: 0.667

Precision: 0.222

Recall: 0.667

Mean reciprocal ranking: 0.444

```python
from ranx import Qrels, Run, evaluate

# Expectations
qrels_dict = {
    "query_1": {"doc_12": 1},
    "query_2": {"doc_14": 1},
    "query_3": {"doc_15": 1}
}

qrels = Qrels(qrels_dict)

# Runs or results
runs_dict = {
    "query_1": {"doc_12": 0.9, "doc_13": 0.8, "doc_19": 0.7},
    "query_2": {"doc_21": 0.7, "doc_29": 0.7, "doc_30": 0.2},
    "query_3": {"doc_13": 0.7, "doc_17": 0.4, "doc_15": 0.2},
}

run = Run(runs_dict)
```

# Retriever evaluation example

## LLM evaluation

- In this case there are no expected documents and you ask the LLM to evaluate the results by relevance to the question and give a score.
- You then take the average of all scored documents

**Prompt:**

First, score each document on an integer scale of 0 to 2 with the following meanings:
0 = represents that the document is irrelevant to the question and cannot be used to answer the question.
1 = represents that the document is somewhat relevant to the question and contains some information that could be used to answer the question
2 = represents that the document is highly relevant to the question and must be used to answer the question.

The user will provide the context, consisting of multiple documents wrapped in document id tags, for example - <doc_1>, <doc_2>, etc
…
Do not provide any code in the result. Provide each score in the following JSON format:
{"final_scores":[{"id": <doc_#>, "relevance":<integer_score >}, ...]]}

Source: www.wandb.courses

# Response evaluation

## Evaluation based on NLP metrics

Classic NLP text comparison metrics:

- Similarity ratio, Levenshtein ratio, ROUGE-L F1 score, BLEU score.

**Note:** this type of metrics does not evaluate semantics and are sometimes hard to interpret.

# Response evaluation

## LLM as response judge

Uses the LLM to score documents based on their relevance to a given question.

Does not require the creation of a dataset.

Criteria to evaluate:

- Correctness
- Relevancy
- Factfullness

# Response evaluation

## Human evaluation

- Having a human team evaluating the results is also a very good, but costly strategy.
- You can have a direct or pair-wise evaluation of the answers.

# Response evaluation example

## NLP metrics

In this case we compare two responses and get numeric scores:

Similarity ratio: 0.263

Levenshtein ratio: 0.397

Rouge score: 0.295

Bleu score: 0.084

**Text 1:**
In Python programming, we can enhance error handling by directing output to the standard error stream. This is achieved by using the `print()` function with the keyword argument `file=sys.stderr`. Implementing this approach ensures that error messages are appropriately separated from standard output, contributing to more robust and reliable applications.

**Text 2:**
In Python, you can use the print() function to print to stderr by passing the value file=sys.stderr as a keyword argument.

# Response evaluation example

## LLM evaluation

You can evaluate multiple criteria like:

- Correctness
- Relevance
- Factuality

The input in this case will be:

- Query
- Reference Answer
- Answer to be scored

LLM not deterministic, so score multiple times and use averages

**Prompt:**

You are tasked with judging the correctness of a generated answer based on the user's question and a reference answer.

You will be given the following information:
1. question
2. reference answer
3. agent answer

Important Instruction: To evaluate the generated answer, follow these steps:

1. Intent Analysis: Consider the underlying intent of the question.
2. Relevance: Check if the generated answer addresses all aspects of the question.
3. Accuracy: Compare the generated answer to the reference for completeness and correctness.
4. Trustworthiness: Measure how trustworthy the generated answer is compared to the reference.

Assign a score on an integer scale of 0 to 2 with the following meanings:
- 0 = The generated answer is incorrect and does not satisfy any criteria.
- 1 = The generated answer is partially correct, contains mistakes, or is not factually correct.
- 2 = The generated answer is correct, thoroughly answers the question, contains no mistakes, and is factually consistent with the reference answer.

Source: www.wandb.courses

# GraphRAG

# Basic RAG limitations (1 of 2)

## Not good for global questions

Basic RAG performs a local topic search, so it will address well-localised questions, but not more “global” questions, like e.g.:

- “What are the main topics in this book?”
- “What are we dealing with in this dataset?”

## No top to bottom answers

- Basic RAG tends to give direct answers without touching the overarching topic.
- If you want answers constructed based on main topics that dive into the specifics, Basic RAG will not be suited.

# Basic RAG limitations (2 of 2)

## Specific answers

- The answers do often not touch related topics.
- So, if you ask a question about tents, it will retrieve text about tents, but overarching topics like “camping”, “holidays”, and “travel” are probably not mentioned.

# What is GraphRAG?

- GraphRAG by Microsoft addresses the problem of generic questions that normal RAG does not address well.
- It uses the output of a Knowledge Graph based on a text corpus to create a much better search experience.
- It also supports different ways of performing RAG queries.

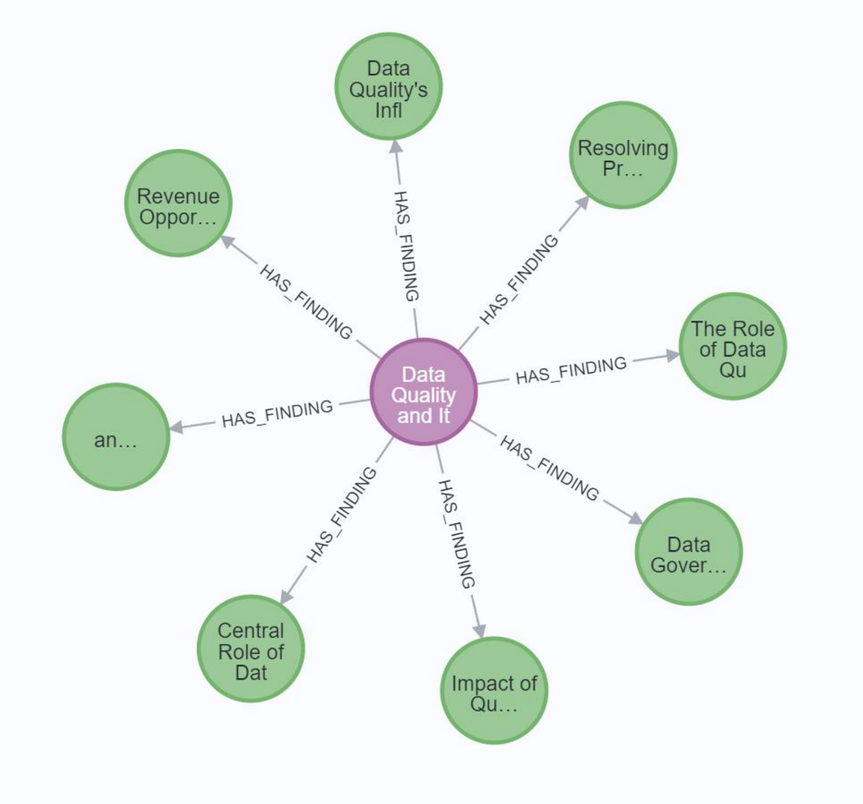
GraphRAG search operates in two modes

Global search

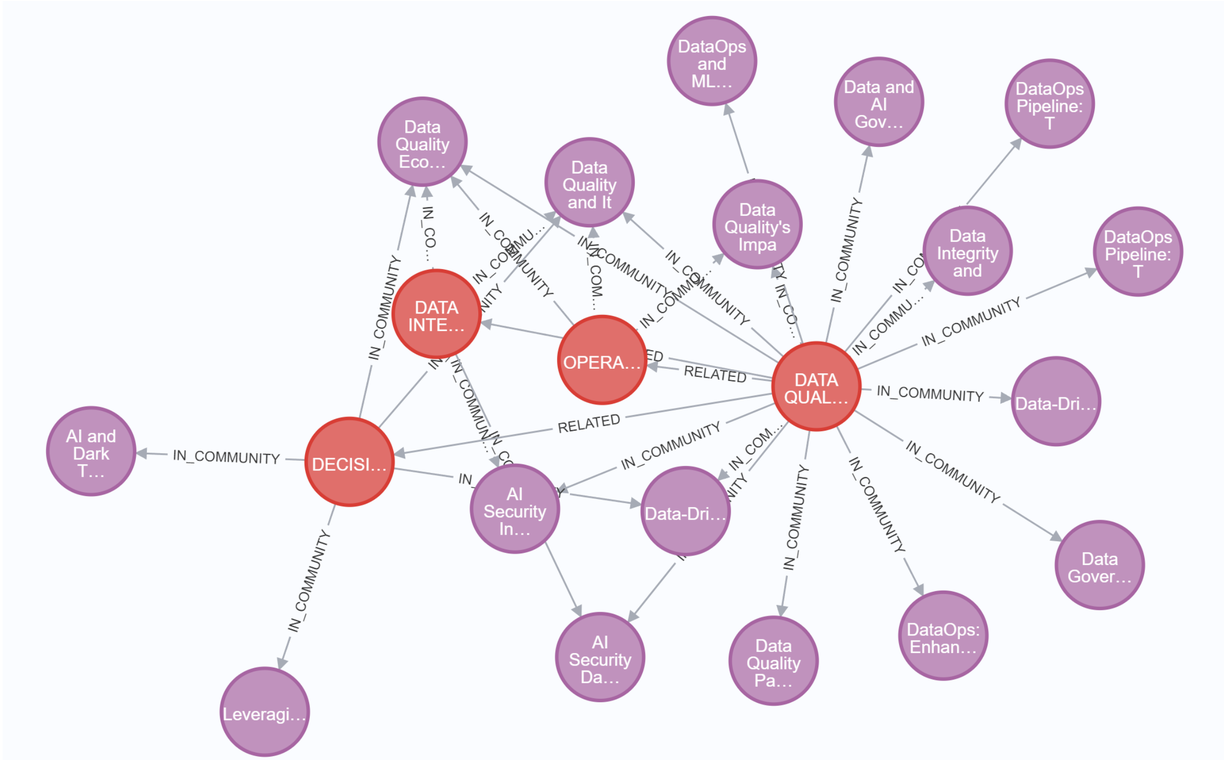
Local search

# What is GraphRAG?

MATCH p=()-[r:HAS_FINDING]->() RETURN p LIMIT 25

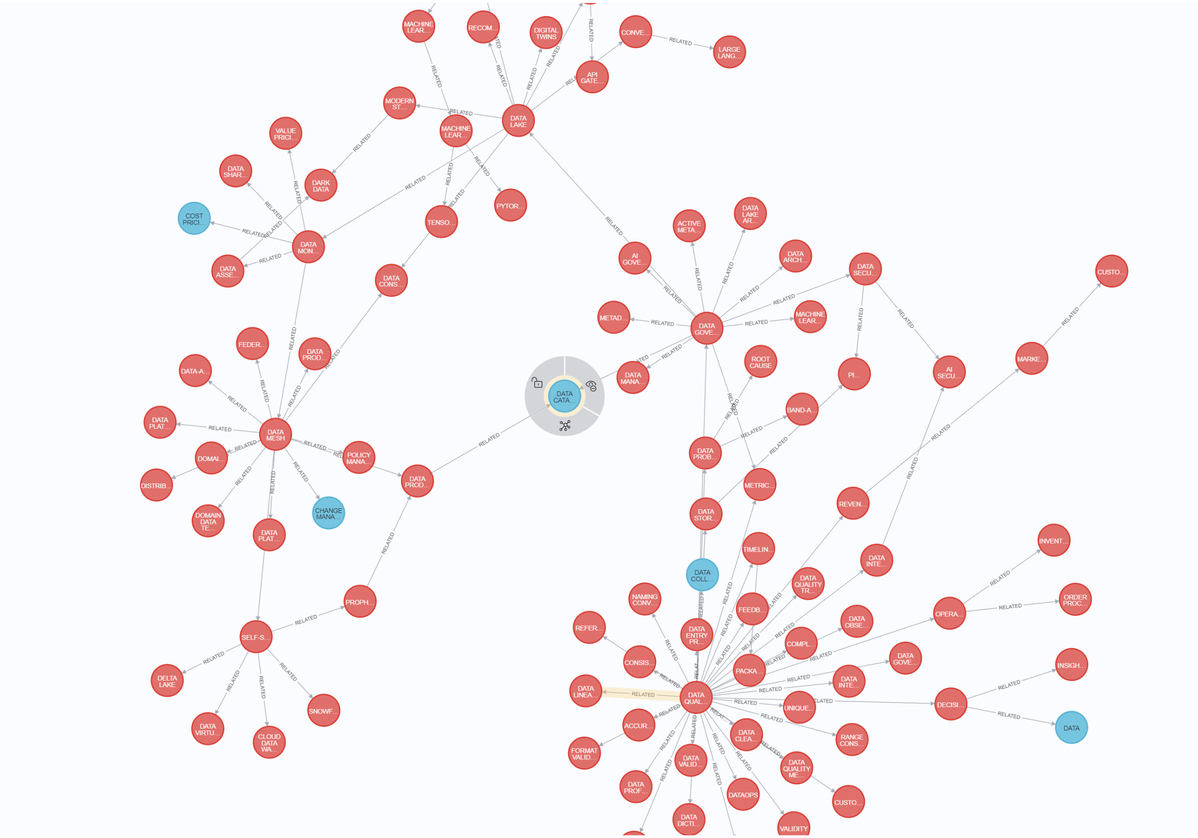


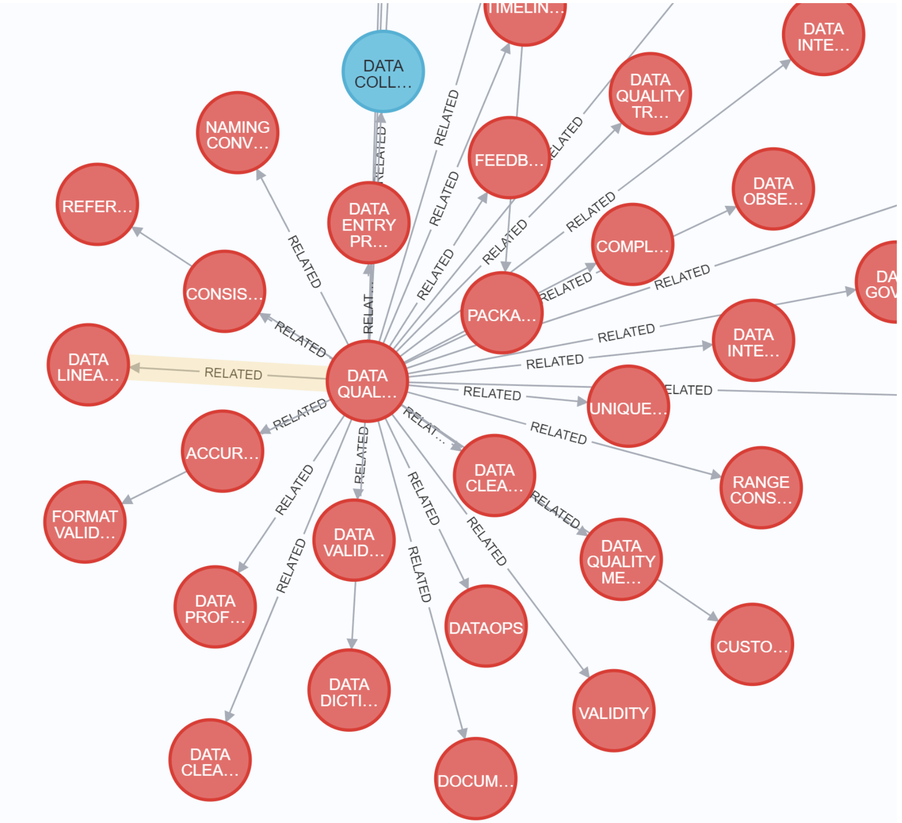

MATCH p=()-[r:IN_COMMUNITY]->() RETURN p LIMIT 25



# What is GraphRAG?

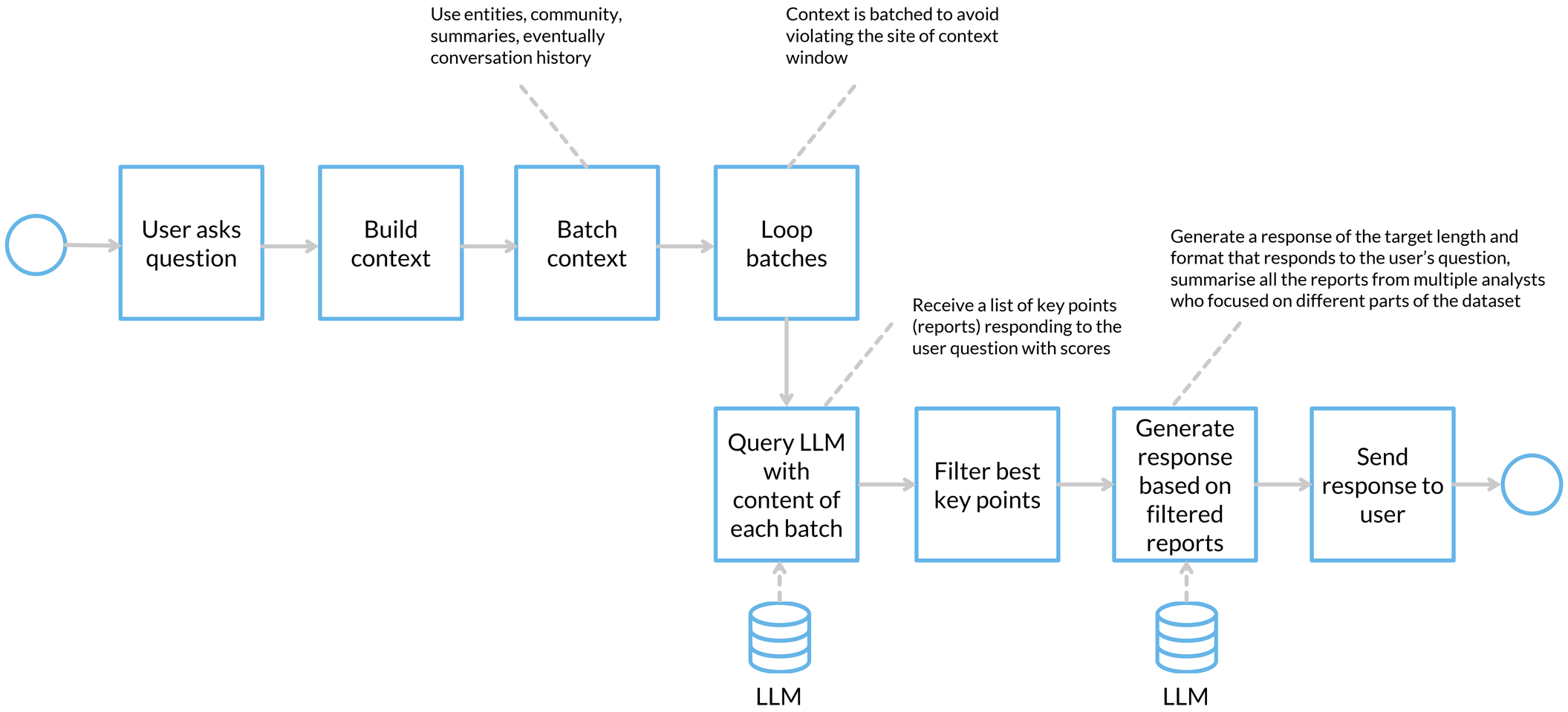
MATCH p=()-[r:RELATED]->() RETURN p LIMIT 100



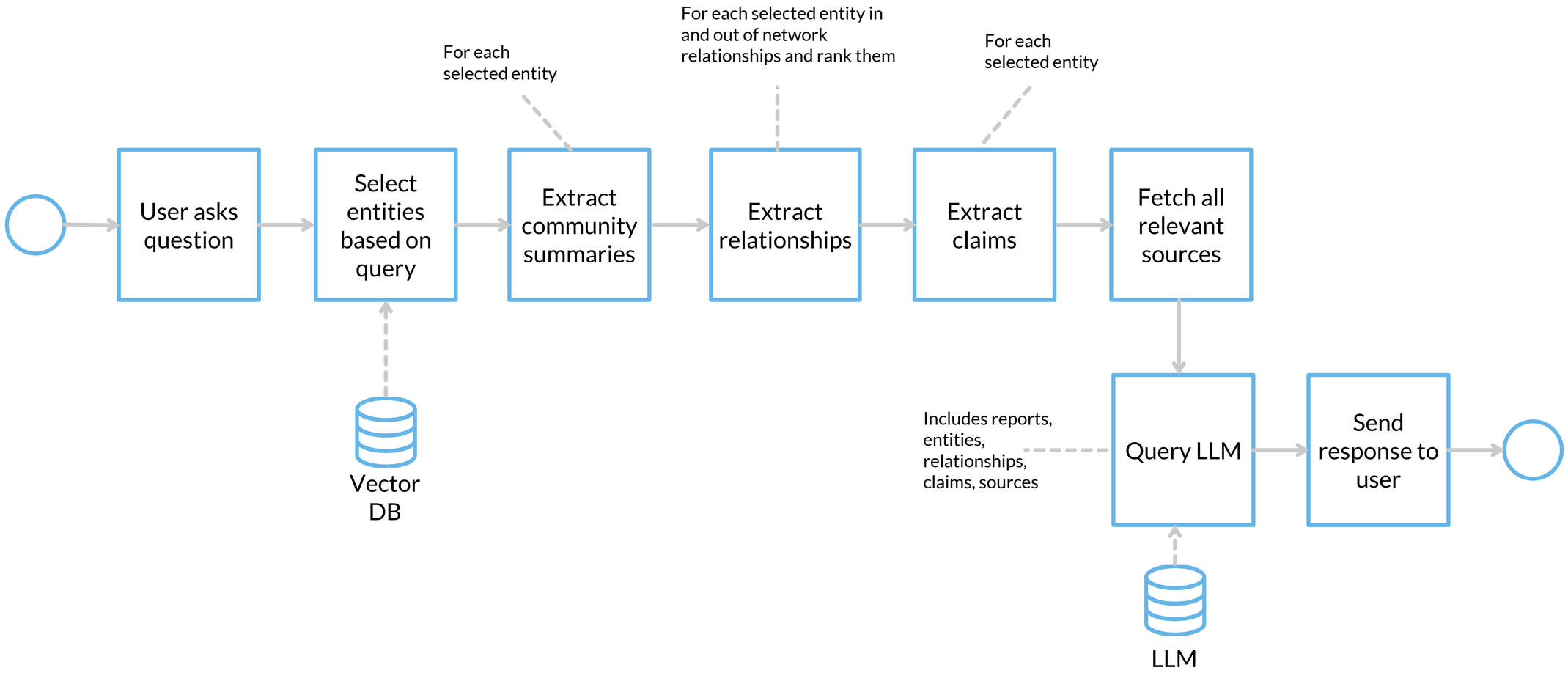


GraphRAG

# Global search



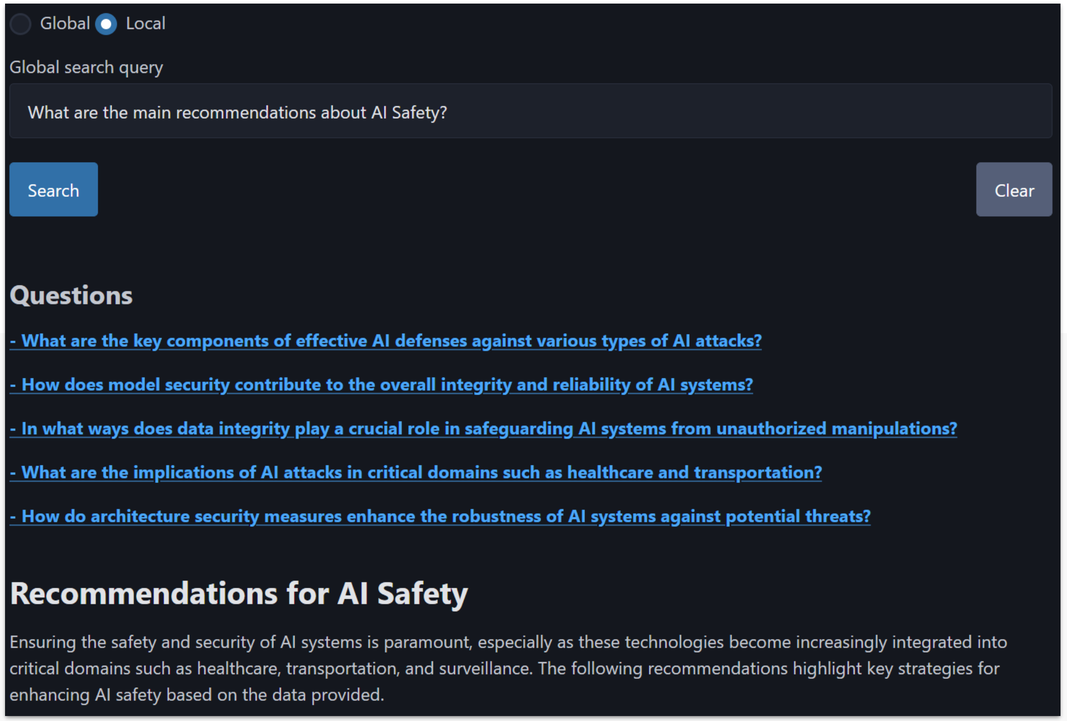
# Local search



# GraphRAG as question generator

## Prompt generator

- GraphRAG can also be used to generate questions based on a history of queries.
- GraphRAG question generation happens in local mode



# GraphRAG as generator

## Prompt generator for knowledge graph

- GraphRAG can also be used to generate questions based on a history of queries.
- GraphRAG question generation happens in local mode

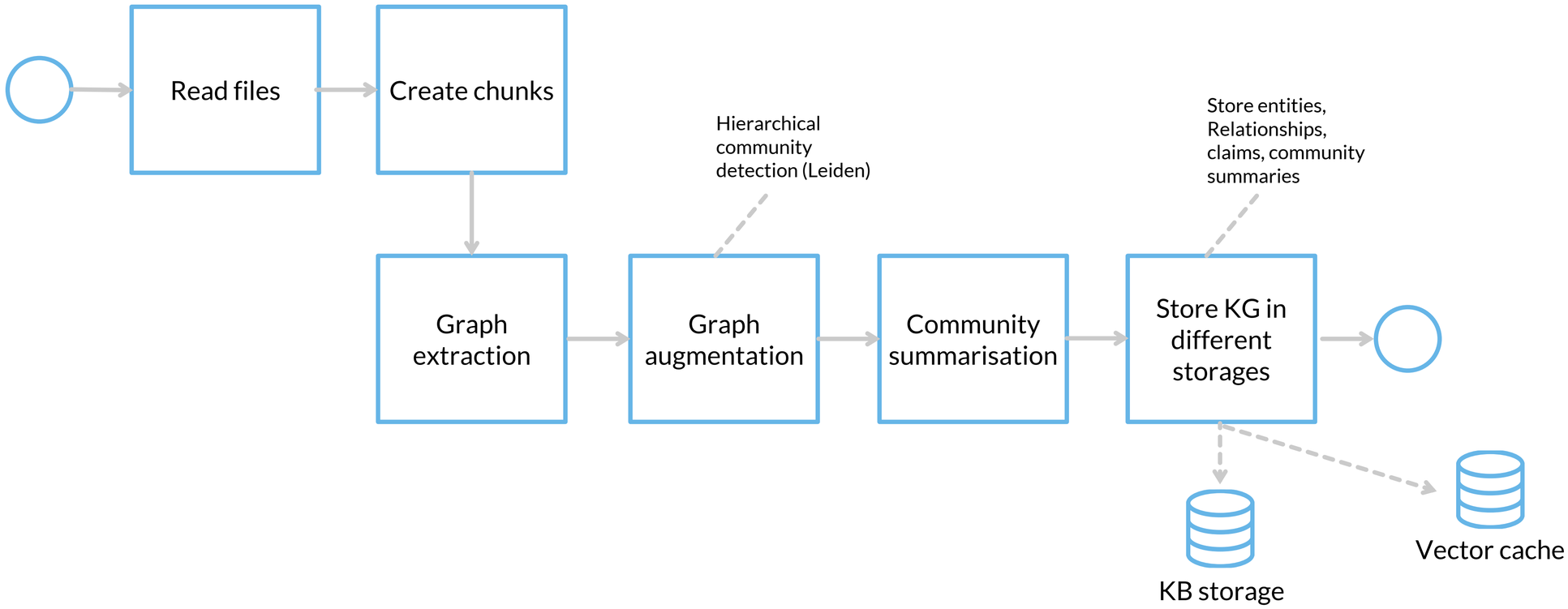
**community.report prompt**:
*You are an expert in Data Management and Security. You are skilled at analysing complex data structures, understanding security protocols, and identifying the interrelations within digital communities. You are adept at helping people navigate the intricacies of data governance, ensuring secure data practices, and mapping out the structure and relationships within the Data Management and Security domain.*

*# GoalWrite a comprehensive assessment report of a community taking on the role of a Data Governance and Security Analyst*
…

**summarise_descriptions prompt**:
*You are an expert in Data Management and Security. You are skilled at analysing complex data structures, understanding security protocols, and identifying the interrelations within digital communities. You are adept at helping people navigate the intricacies of data governance, ensuring secure data practices, and mapping out the structure and relationships within the Data Management and Security domain.*
*Using your expertise, you're asked to generate a comprehensive summary of the data provided below.*
…

# GraphRAG index workflow

GraphRAG creates a knowledge graph and also indexes part of its content in a vector database.



# Incorporating knowledge into LLMs

# Audience poll

In which areas would your company profit from information retrieval?

# Main idea

- RAG is efficient to extend the knowledge of an LLM, but the knowledge is kept outside of the LLM in a different system.
- The main idea is to have the external knowledge to be part of the LLM itself.

These are the main options to incorporate external knowledge in LLMs:

Model fine tuning

Long context models

Memory transformer

# Model fine tuning

Model fine tuning requires creating your own dataset and fine tuning the weights of the upper layers of the model on this dataset.

- Integrates the external knowledge successfully into the fine-tuned model

- Requires fine tuning which is resource intensive
- On every update of the knowledge base, you have to fine tune again
- Slow to update

# Long memory transformers

Models with a very large context window, which can have millions of tokens.

- Models have a huge context window (e.g Gemini Pro 1.5 with 2 million) and can process in one go lots of information

- Expensive to use
- Copying large contexts on every request is potentially slow.
- Lots of redundancy on specific questions
- Risks confusion on answering

# Memory transformer / Extended mind transformer

The main idea is about reading and memorising data at inference time without training the model or creating large context windows. The retrieval mechanism for the relevant facts is internal to the model.

- Model memory easy to update
- Retrieval internal to the model, so really fast
- Able to retrieve the memories which the model found useful

- Fairly new and untested
- You will need to run the model yourself

# Extended mind transformer basic ideas (1 of 2)

This is how the extended mind transformer operates

**Memory injection**

- After instantiating the model, you index your knowledge base.
- You inject the memories as tokens directly into the model.

# Extended mind transformer examples (1 of 2)

```
[ ]  from transformers import AutoModelForCausalLM, AutoTokenizer

     MODEL = "normalcomputing/extended-mind-llama-2-7b-chat"
     tokenizer = AutoTokenizer.from_pretrained(MODEL)
     model = AutoModelForCausalLM.from_pretrained(MODEL, trust_remote_code=True).to(device)
```

```
/usr/local/lib/python3.10/dist-packages/huggingface_hub/utils/_token.py:89: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens),
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(
pytorch_model.bin.index.json: 100% 23.9k/23.9k [00:00<00:00, 1.89MB/s]
Downloading shards: 100% 3/3 [08:57<00:00, 172.64s/it]
```

```
     Employees should not display tattoos that could cause offence and employees who are
     client/customer-facing, or in specific roles, may be asked to cover up tattoos."""
```

```
[ ]  memories = tokenizer(about_onepoint_entry).input_ids
     model.memory_ids = memories
```

# Extended mind transformer examples (2 of 2)

## Inference

```
[ ]  def generate_inputs(query: str) -> torch.Tensor:
       inputs = tokenizer(query, return_tensors="pt").input_ids
       return inputs
```

```
[ ]  def execute_model(inp: str, max_length: int = 100) -> str:
       inputs = generate_inputs(inp)
       outputs = model.generate(inputs.to(device), max_length=max_length, topk=2, output_retrieved_memory_idx=True, output_attentions=True, return_dict_in_generate=True,)
       return tokenizer.decode(outputs['sequences'][0], skip_special_tokens=True), outputs
```

```
print(execute_model("What kind of company is Onepoint Consulting?"[0], 1000))
```

```
What kind of company is Onepoint Consulting?
Onepoint Consulting is an enterprise architecture consulting company that provides services to organizations looking to improve their IT

Onepoint Consulting is a niche consulting company that specializes in enterprise architecture consulting. They work with organizations to

Onepoint Consulting has a horizontal management structure, which means that major and strategic decisions regarding human resources, busi

Overall, Onepoint Consulting is a specialized consulting company that helps organizations navigate the complex world of enterprise archit
```

# Extended mind transformer basic ideas (2 of 2)

This is how the extended mind transformer operates

## Inference

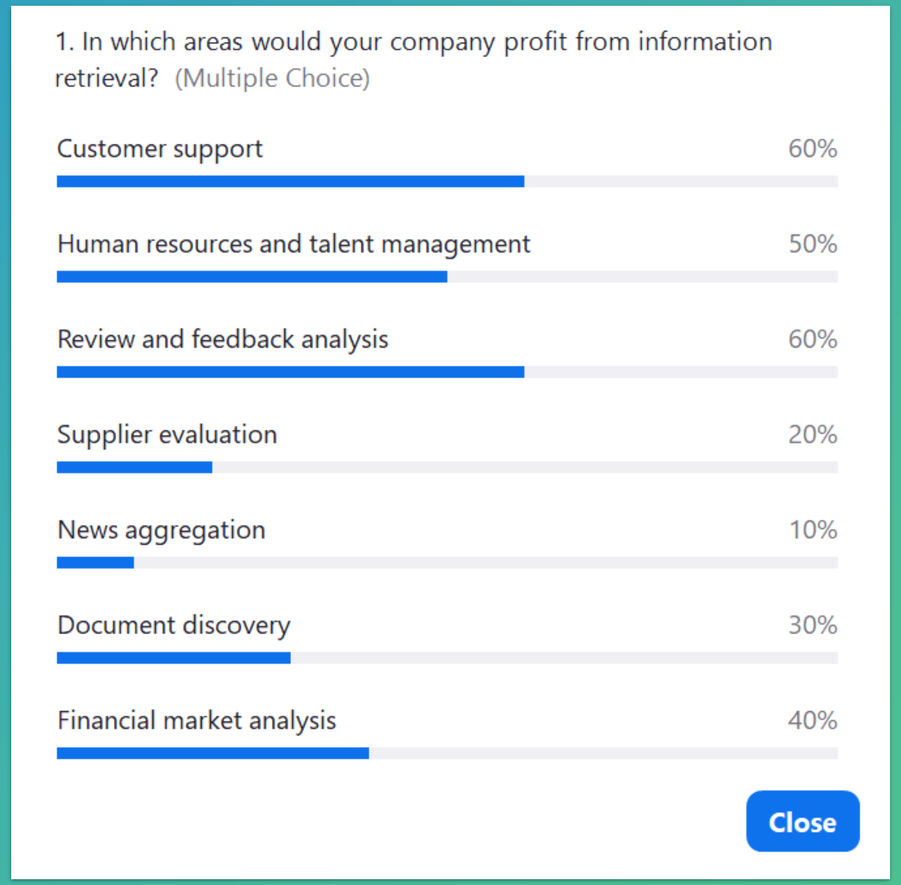
### Lookup

- The input tokens are used to perform a lookup in the internal memories using cosine similarity (a distance measurement).
- This retrieves a representation of the memories closest to the query.

### Injection into attention layers

- The retrieved memories are applied on each attention block of the decoder layer of the transformer, affecting the output

# Audience insights



# Credits

chatgpt.com

## GraphRAG

https://microsoft.github.io/graphrag/

## Extended mind transformers

github.com/normal-computing/extended-mind-transformers

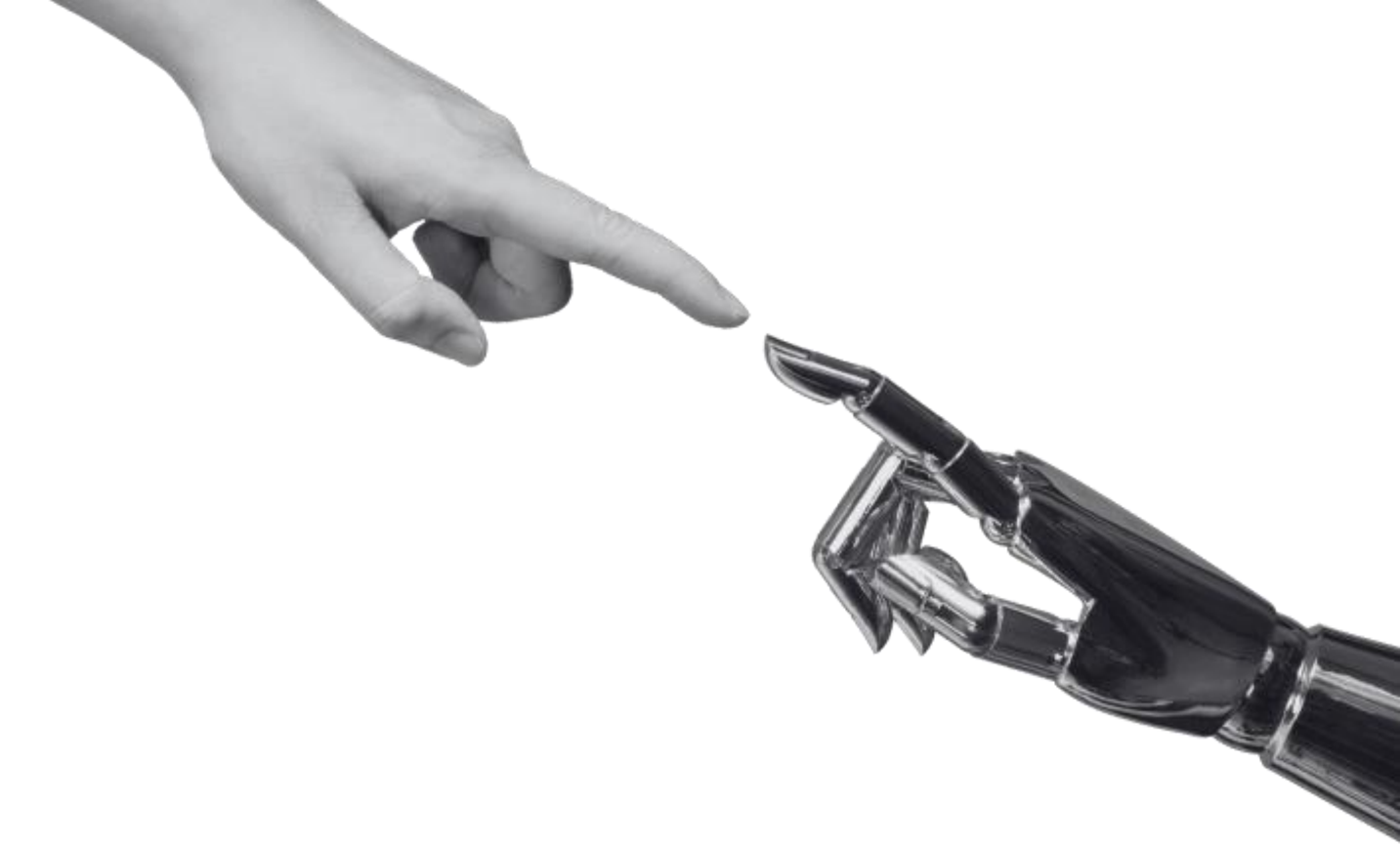
## Wandbot

github.com/wandb/wandbot

## LangChain

langchain-ai.github.io/langgraph/tutorials/rag/langgraph_agentic_rag

# Thank you for joining



Please feel free to contact Gil Fernandes if you have any feedback about the session.

Email at techtalk@onepointltd.com

Connect on LinkedIn www.linkedin.com/in/gil-palma-fernandes

Find Gil's Reflections on AI at medium.com/@gil.fernandes

Onepoint
TechTalk

Session 6

# AI agents

Tuesday, 21 January 2025 | 11:00 UK | 12:00 CEST

Our awesome speaker

Gil Fernandes
AI Solutions Engineer
Onepoint

**Fruit**

3. How to turn the spotlight on Dark Data with Gen AI

**Leaves**

6. **AI agents**
7. Ensemble of agents
8. Building complex simulations
9. Building LLMs based systems

**Branches**

4. Workflows and complex interactions
5. RAG, GraphRAG and beyond

**Trunk**

2. Direct interactions with LLMs

**Roots**

1. Fundamentals of LLMs

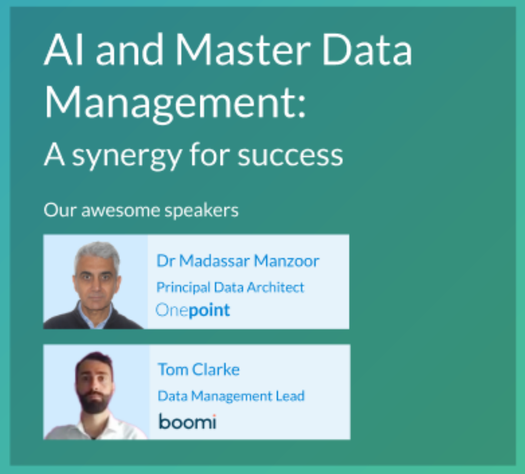
Onepoint
TechTalk

# Missed previous webinars?

Watch the replays at onepointltd.com/techtalk

## Recent webinars





Onepoint is a **boutique, values-driven, business-oriented technology consultancy**.

We **architect, prototype, build, and manage** data and AI powered solutions. We partner with global clients looking for high-impact, enterprise-grade advice and IT services to realise their most critical digital transformations.

London | Manchester | Pune

Onepoint

*Your trusted companions for the digital journey™*

onepointltd.com

hello@onepointltd.com