

Retrieval-Augmented Generation (RAG) for LLMs

Concepts & Techniques for building efficient RAG systems

Sarath Chandra on 17th Jan 2024


What & Why RAG?

RAG is a process where external data is *retrieved* and then passed to the LLM when doing the *generation* step.

SA

You

What is LangSmith?



ChatGPT

As of my last knowledge update in January 2022, I don't have specific information about something called "LangSmith." It's possible that there have been developments or new information since then. If "LangSmith" refers to a technology, product, person, or concept that emerged after that date, I recommend checking the latest sources such as news articles, company websites, or other reliable resources for the most up-to-date information. If "LangSmith" is a common term or a less-known entity, providing additional context might help me better understand and provide relevant information.

From ChatGPT

Microsoft Bing

SEARCH

CHAT

Sarath

1019

Choose a conversation style

More Creative

More Balanced

More Precise

What is LangSmith?


✓ Searching for: **LangSmith**


✓ Generating answers for you...


LangSmith is an all-in-one toolkit for building and deploying language learning models (LLM) applications with confidence ¹. It offers features such as trace logging, prompt evaluation, feedback collection, and AI-assisted evaluation for every step of the application lifecycle ¹. LangSmith emerges as a robust and versatile testing framework for evaluating language models and AI applications ². It is capable of building production-grade LLM applications and seamlessly integrates with LangChain, the go-to open-source framework for building with LLMs ³.

I hope this helps! Let me know if you have any other questions.

Learn more

1  langchain.com

2  blog.logrocket.com

3  infoworld.com

1 of 30

?

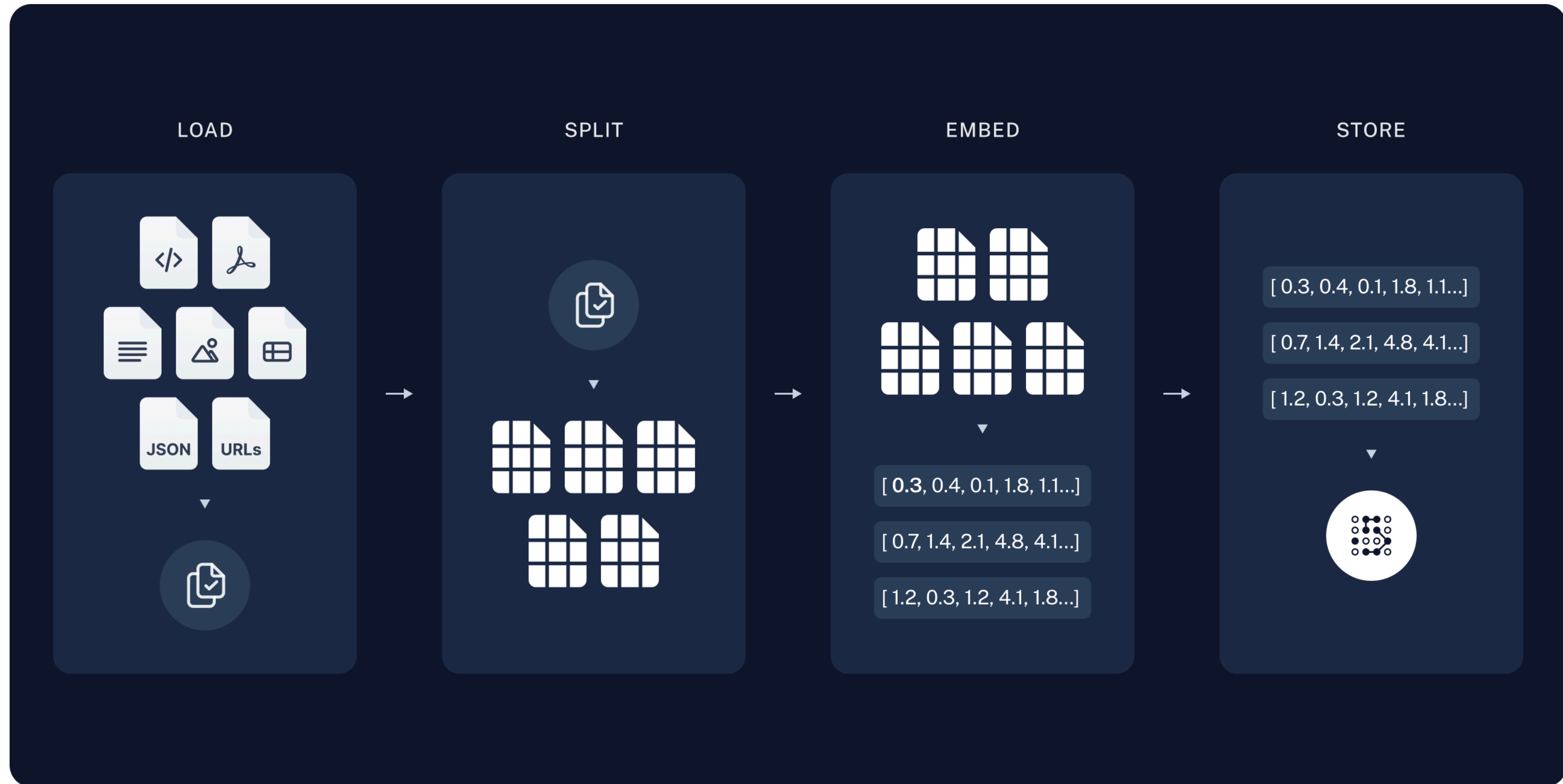
What is LangChain?

How does LangSmith work?

Can I use it for free?

RAG Step 1: Indexing

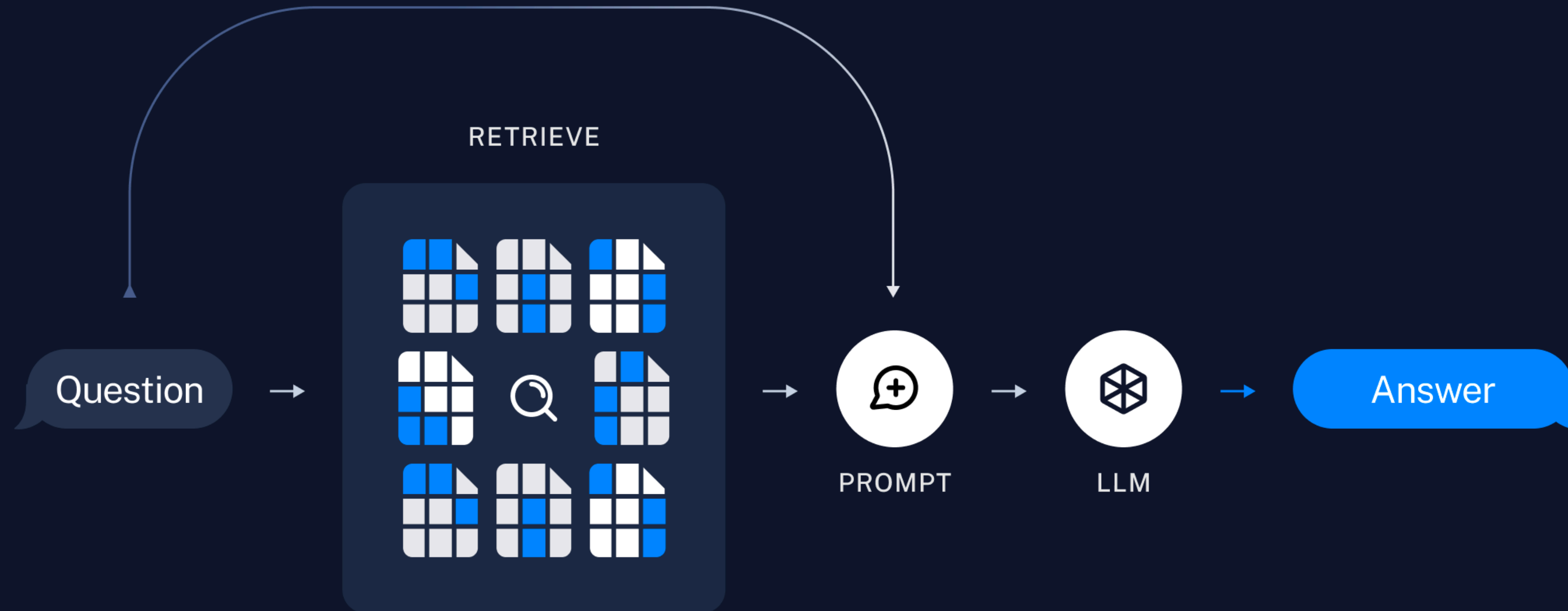
Indexing is a pipeline for ingesting data from a source and indexing it. *This usually happens offline.*



- DocumentLoaders Text splitters Embeddings. VectorStore

RAG Step 2: Retrieval and generation

The actual RAG chain, which takes the user query at run time and retrieves the relevant data from the index, then passes that to the model



Advanced Retrievers

Vector store-backed retriever

```
embeddings = OpenAIEmbeddings(),  
db = FAISS.from_documents(texts, embeddings)
```

```
retriever = db.as_retriever()
```

```
docs = retriever.get_relevant_documents("what did he say")
```

Maximum marginal relevance retrieval

```
retriever = db.as_retriever(search_type="mmr")
```

```
docs = retriever.get_relevant_documents("what did he say about ketanji brown jackson")
```

Similarity score threshold retrieval

```
retriever = db.as_retriever(  
    search_type="similarity_score_threshold", search_kwargs={"score_threshold": 0.5}  
)
```

```
docs = retriever.get_relevant_documents("what did he say about ketanji brown jackson")
```

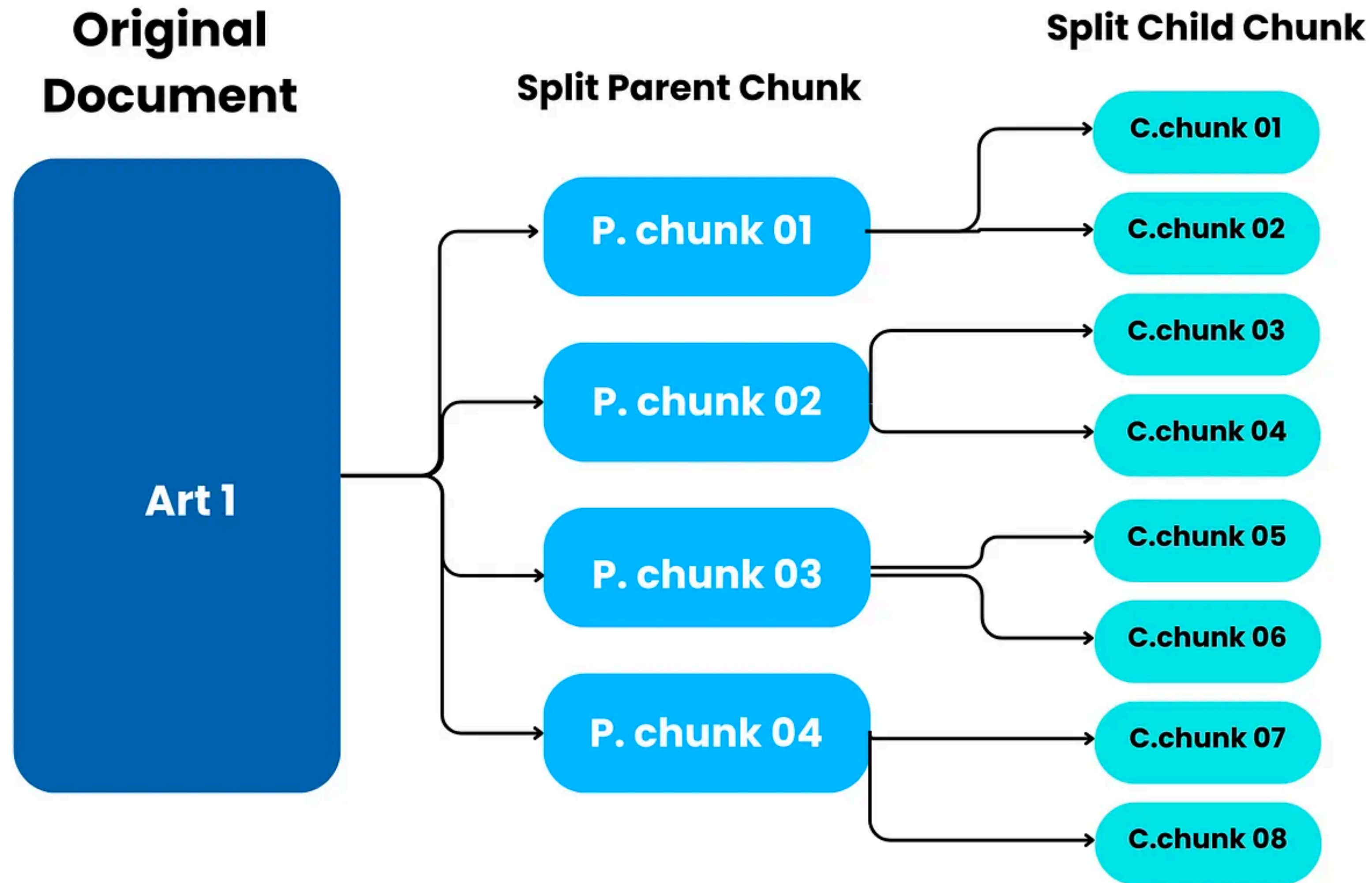
Specifying top k

```
retriever = db.as_retriever(search_kwargs={"k": 1})
```

```
docs = retriever.get_relevant_documents("what did he say about ketanji brown jackson")  
len(docs)
```

Advanced Retrievers

ParentDocument



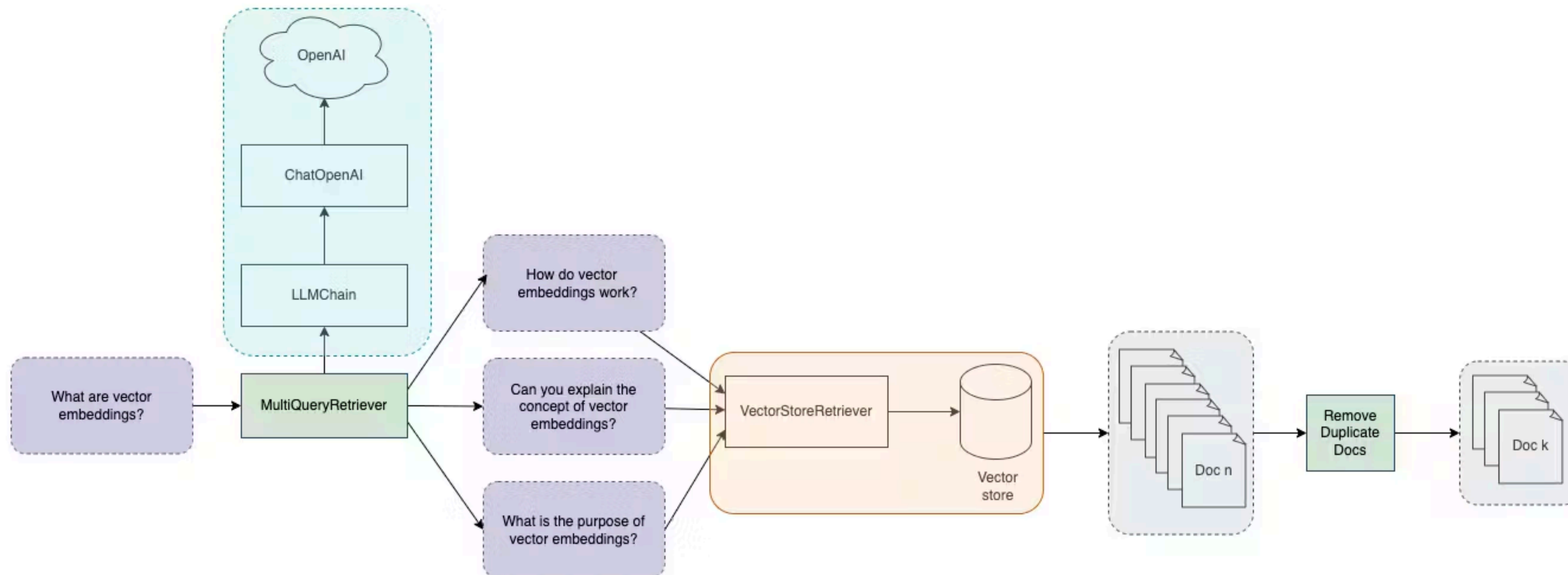
- Conflicting desires when splitting
- Small documents and their embeddings can most accurately reflect their meaning. If too long, then the embeddings can lose meaning
- We want to have long enough documents that the context of each chunk is retained.

Advanced Retrievers

MultiQueryRetriever

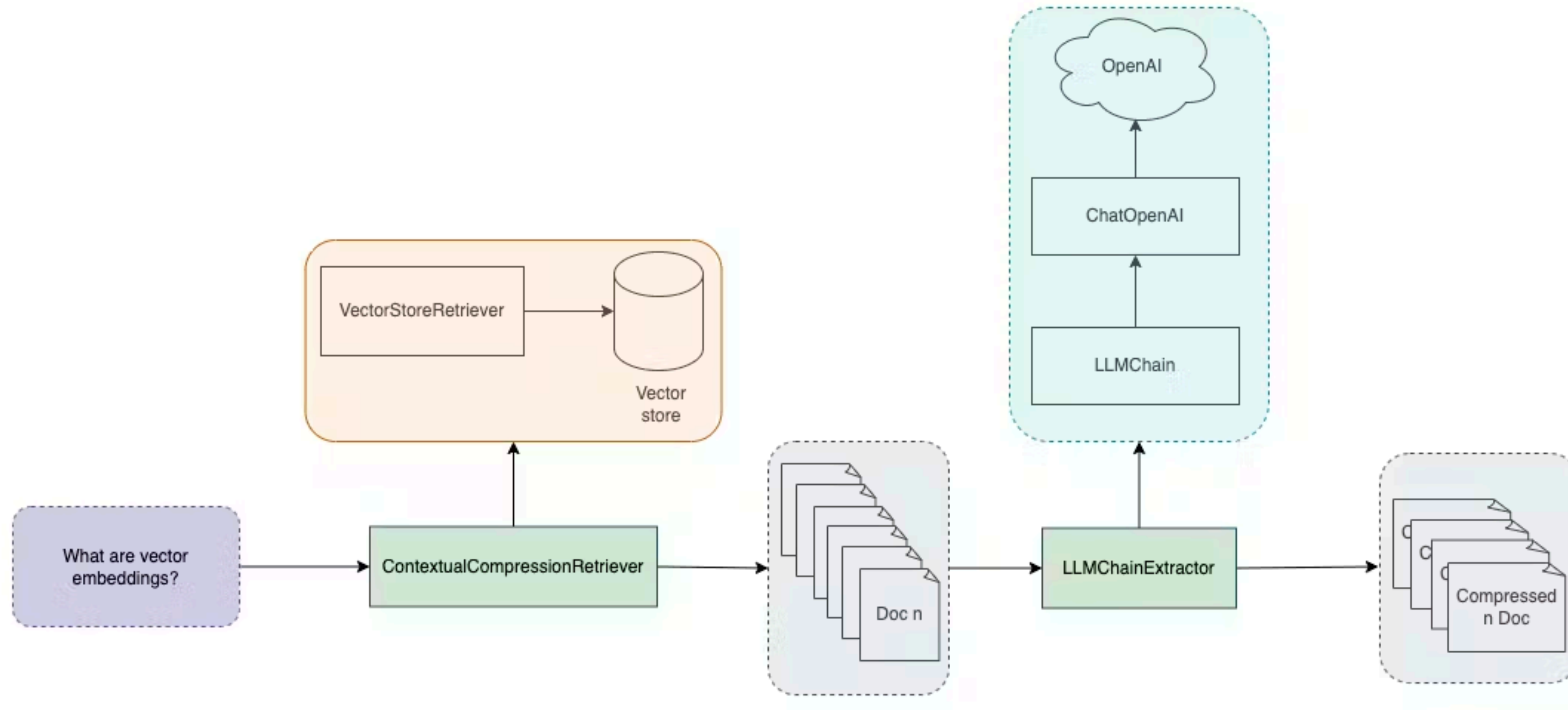
- **Prompt:** You are an AI language model assistant. **Your task is to generate three different versions of the given user question to retrieve relevant documents from a vector database.** By generating **multiple perspectives on the user question**, your goal is to help the user overcome some of the limitations of the **distance-based similarity search**.

Deep Dive into the Internals of Langchain Vector Store Retriever



Advanced Retrievers

Contextual Compression



- Prompt: Given the following question and context, **extract any part of the context *AS IS* that is relevant to answer the question.** If none of the context is relevant return **NO OUTPUT.** Remember, ***DO NOT*** edit the extracted parts of the context.

Advanced Retrievers

Ensemble

```
# initialize the bm25 retriever and faiss retriever
```

```
bm25_retriever = BM25Retriever.from_texts(  
    doc_list_1, metadatas=[{"source": 1}] * len(doc_list_1)  
)
```

```
bm25_retriever.k = 2
```

BM25 : Syntactic

```
doc_list_2 = [  
    "You like apples",  
    "You like oranges",  
]
```

```
embedding = OpenAIEmbeddings()
```

FAISS : Semantic

```
faiss_vectorstore = FAISS.from_texts(  
    doc_list_2, embedding, metadatas=[{"source": 2}] * len(doc_list_2)  
)
```

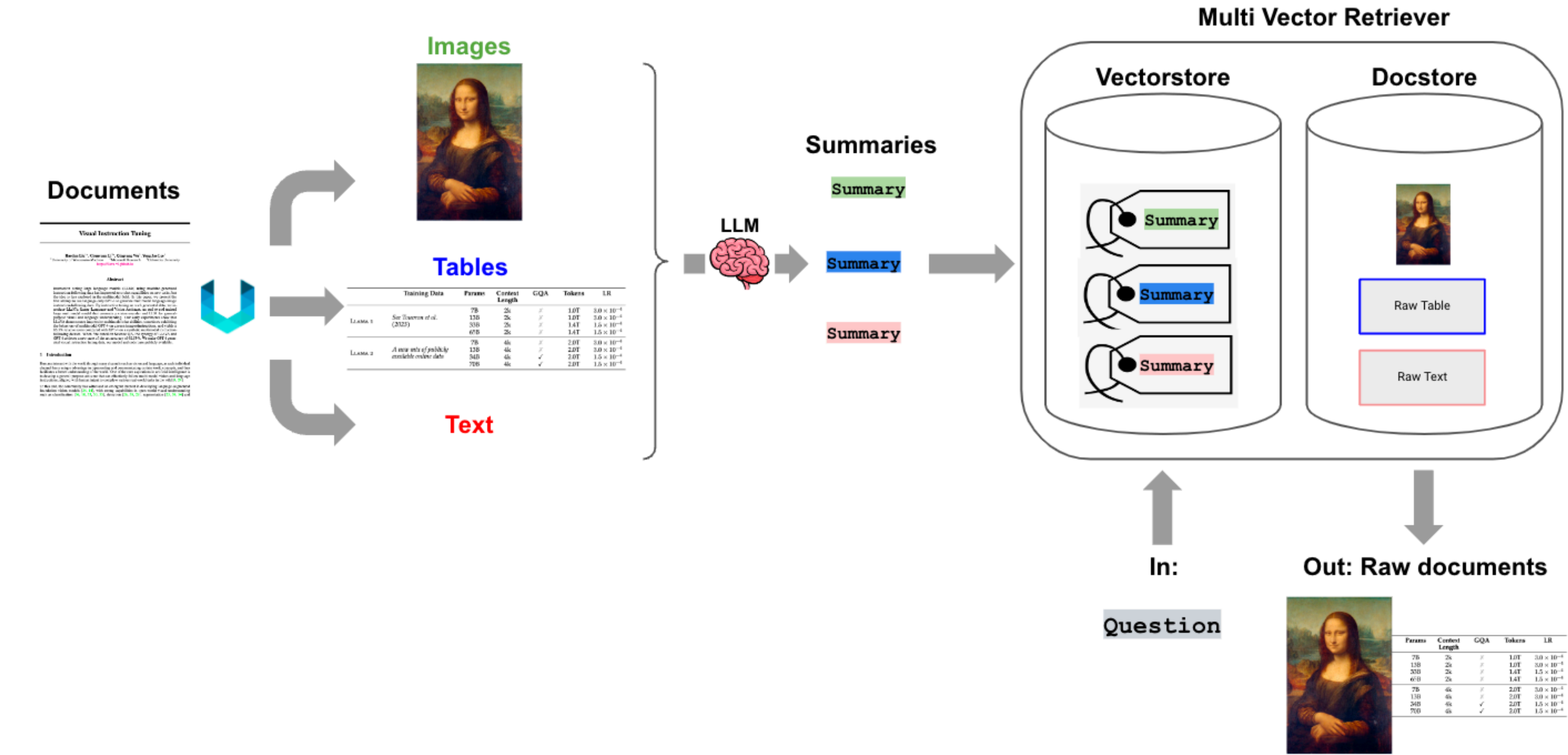
```
faiss_retriever = faiss_vectorstore.as_retriever(search_kwargs={"k": 2})
```

```
# initialize the ensemble retriever
```

```
ensemble_retriever = EnsembleRetriever(  
    retrievers=[bm25_retriever, faiss_retriever], weights=[0.5, 0.5]  
)
```

Advanced Retrievers

Multi Vector



- **Smaller chunks:** split a document into smaller chunks, and embed those (this is ParentDocumentRetriever).
- **Summary:** create a summary for each document, embed that along with (or instead of) the document.
- **Hypothetical questions:** create hypothetical questions that each document would be appropriate to answer, embed those along with (or instead of) the document
- Multi-Vector Retriever for RAG on tables, text, and images

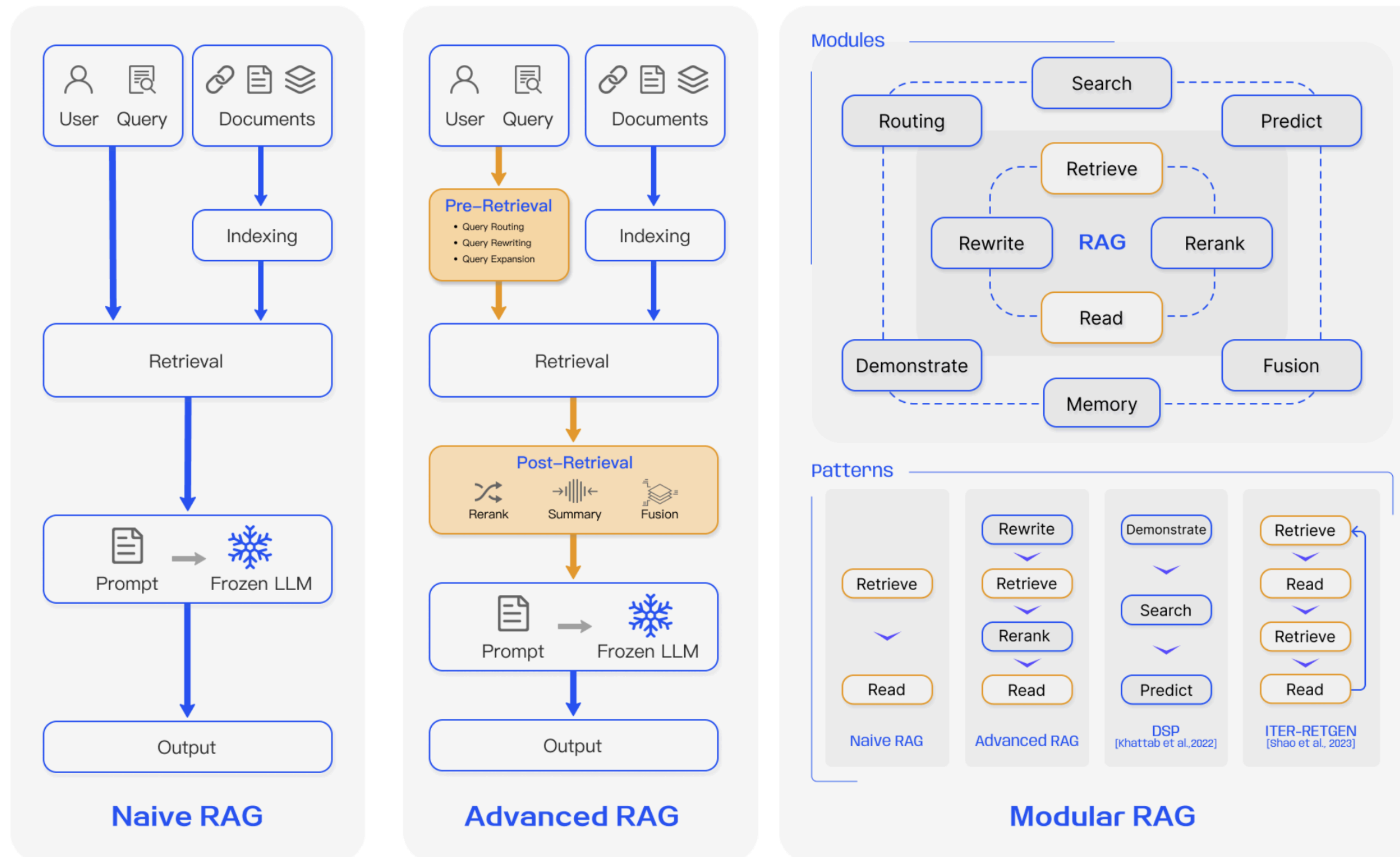
Some more advanced Retrievers

Name	When to Use	Description
Self Query	If users are asking questions that are better answered by fetching documents based on metadata rather than similarity with the text.	This uses an LLM to transform user input into two things: (1) a string to look up semantically, (2) a metadata filter to go along with it. This is useful because oftentimes questions are about the <u>METADATA</u> of documents (not the content itself).
Time-Weighted Vectorstore	If you have timestamps associated with your documents, and you want to retrieve the most recent ones	This fetches documents based on a combination of semantic similarity (as in normal vector retrieval) and <u>recency</u> (looking at timestamps of indexed documents)
Long-Context Reorder	If you are working with a long-context model and noticing that it's not paying attention to information in the middle of retrieved documents.	This fetches documents from an underlying retriever, <u>and then reorders them so that the most similar are near the beginning and end</u> . This is useful because it's been shown that for longer context models they sometimes don't pay attention to information in the middle of the context window.

- Lost in the Middle: How Language Models Use Long Contexts

Assignment

Retrieval-Augmented Generation for Large Language Models: A Survey 18 Dec 2023



- This comprehensive review paper offers a detailed examination of the progression of **RAG paradigms, encompassing the Naive RAG, the Advanced RAG, and the Modular RAG.**
- The paper **highlights** the state-of-the-art technologies embedded in **each of the critical components (which includes the retrieval, the generation and the augmentation techniques)**, providing a profound understanding of the advancements in RAG systems.
- Furthermore, this paper introduces the **metrics and benchmarks for assessing RAG models**, along with the most up-to-date evaluation framework.