# Query Enhancement

## Query Expansion

In a RAG pipeline, the quality of the query sent to the retriever determines how good the retrieved context is — and therefore, how accurate the LLM's final answer will be.

That's where Query Expansion / Enhancement comes in.

**What is Query Enhancement?**

Query enhancement refers to techniques used to improve or reformulate the user query to retrieve better, more relevant documents from the knowledge base.

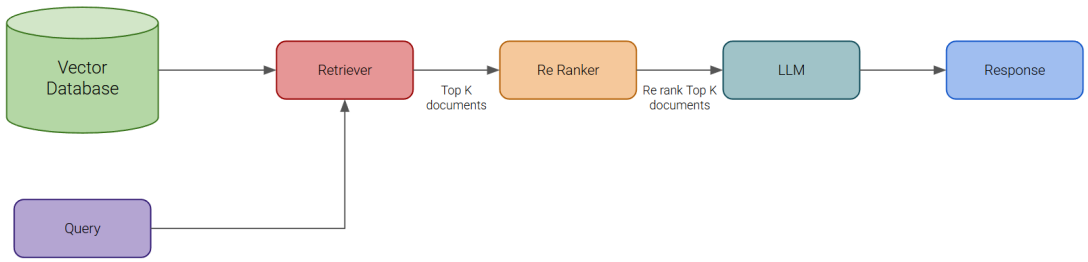
It is especially useful when:

1. The original query is short, ambiguous, or under-specified.
2. You want to broaden the scope to catch synonyms, related phrases, or spelling variants.

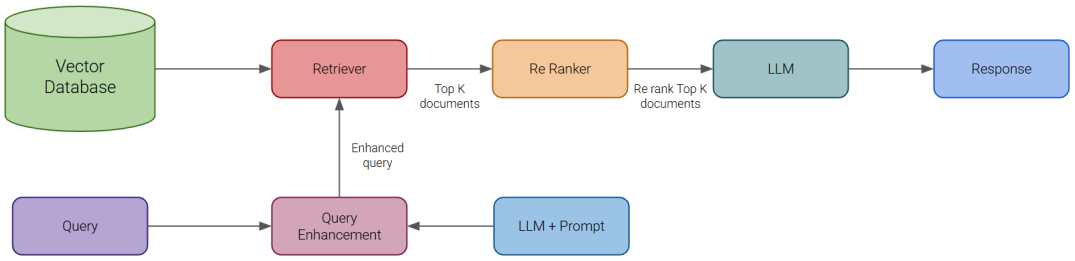
## Why Query Expansion Matters in RAG

| **Problem (Original Query)** | **Solution (Enhanced Query)** |
|---|---|
| LangChain Memory | LangChain memory modules, conversation memory |
| Tools in LLM | LangChain tools, API, calculator, agent tools |
| Retrieval | Vector retrieval, dense search, BM25, MMR |

Better queries → Better retrieved chunks → Better grounded LLM answers




Without Query Expansion



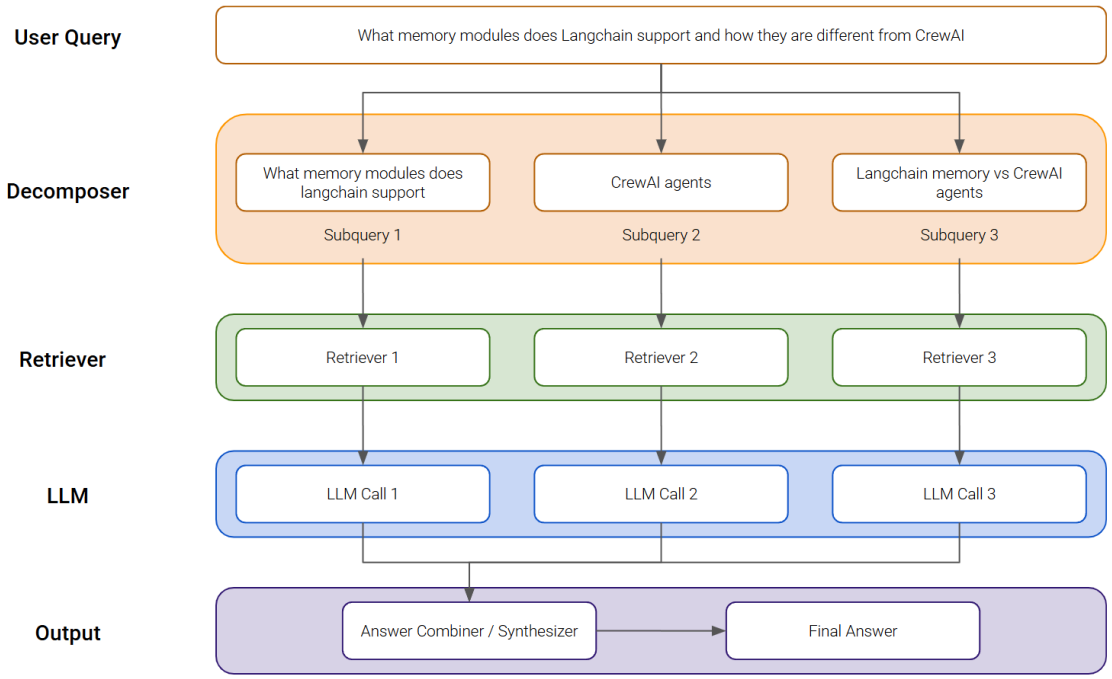

With Query Expansion

# Query Decomposition

Query decomposition is the process of taking a complex, multi-part question and breaking it into simpler, atomic sub-questions that can each be retrieved and answered individually.

## Why Use Query Decomposition

- Complex queries often involved multiple concepts
- LLMs or retrievers may miss parts of the original question
- It enables multi-hop reasoning (answering in steps)
- Allows parallelism (especially in multi-agent frameworks)

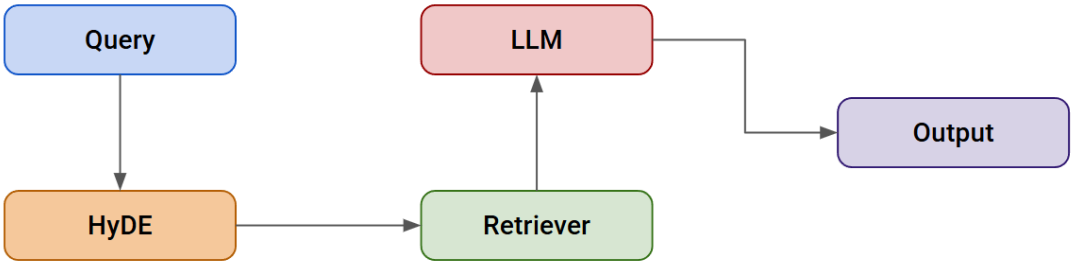


# Hypothetical Document Embedding (HyDE)

HyDE (Hypothetical Document Embedding) is a retrieval technique where, instead of embedding the user's query directly, you first generate a hypothetical answer (document) to the query using an LLM — and then embed that hypothetical document to search your vector store.

HyDE bridges the gap between user intent and relevant content, especially when:

1. Queries are short
2. Language mismatch between query and documents
3. You want to retrieve based on answer content, not question words




## Why Use HyDE

| Problem | How HyDE Helps |
|---|---|
| User query and document use different words | Embeds answer-style content instead of questions |
| Query is to vague | LLM-generated answer give richer semantics |
| Better grounding needed | Embeds what the answer might look like |
| Zero-shot retrieval | Works well even without prior training |

## Benefits of HyDE

| Feature | Why It Helps |
|---|---|
| Semantic intent modeling | Better than literal keyword matching |

| | |
|---|---|
| LLM-aware retrieval | Query is expanded into more context |
| Generalization | Works well even if document phrasing differs from question |
| Plug and play | No need for retraining; works with OpenAI, Cohere, Huggingface |