

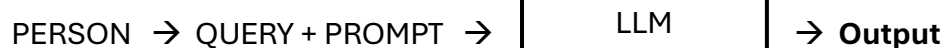
What is RAG:

RAG stands for Retrieval-Augmented Generation.

RAG is the process of optimizing the output of a large language model, so it references an external knowledge base outside of its training data sources before generating a response.

LLM (Large language models) are trained on vast volumes of data and use billions of parameters to generate original output for tasks like – answering questions, translating languages etc. So, RAG extends the already powerful capabilities of LLM models to specific domains or an organizations internal knowledge base, all without the need to retrain the model. It is a cost-effective approach to improving LLM output so it remains relevant, accurate and useful in various contexts.

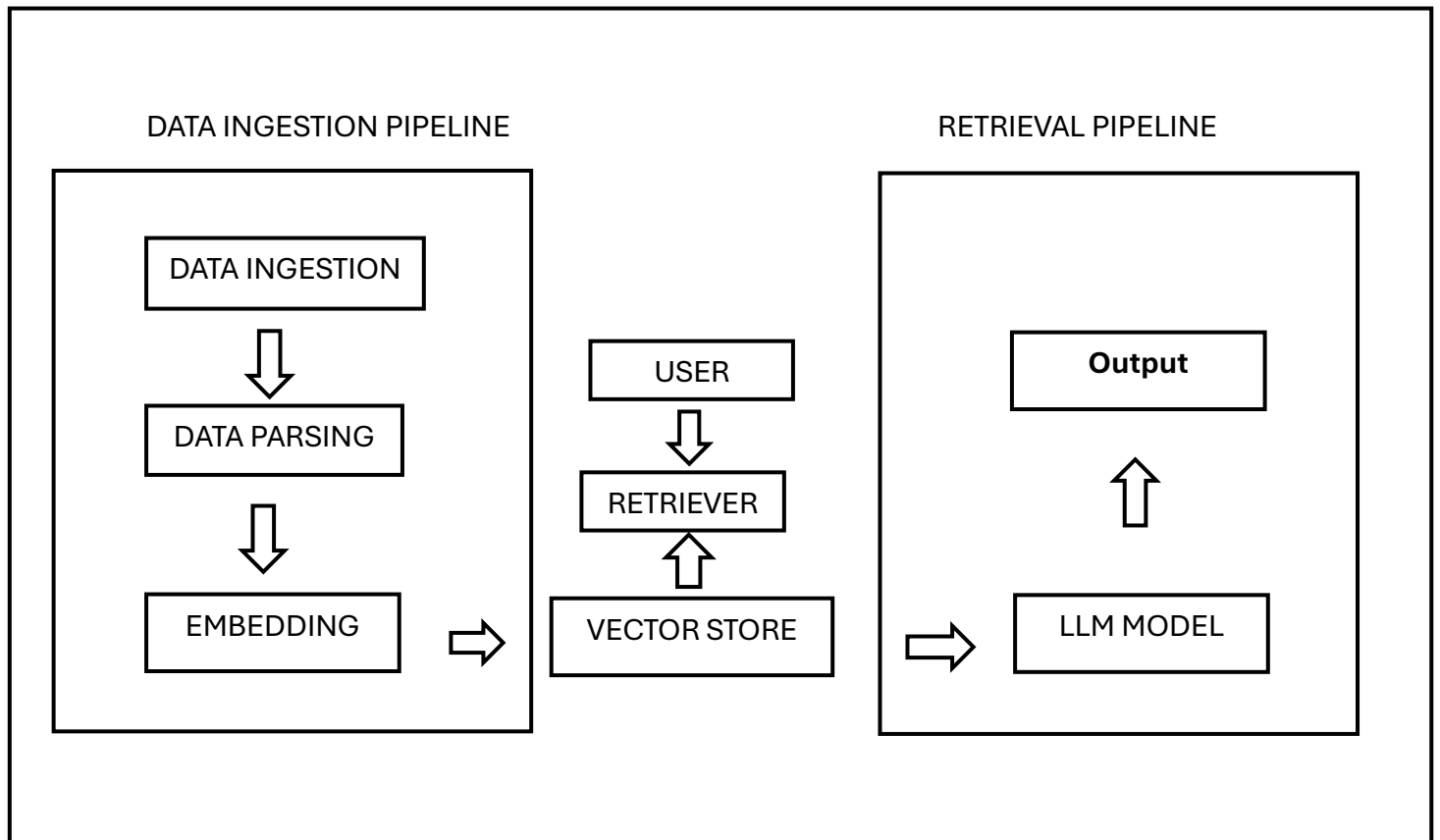
This is the simple flow of any LLM model like ChatGPT:



```
graph LR; PERSON --> QUERY_PROMPT[QUERY + PROMPT]; QUERY_PROMPT --> LLM[LLM]; LLM --> Output
```

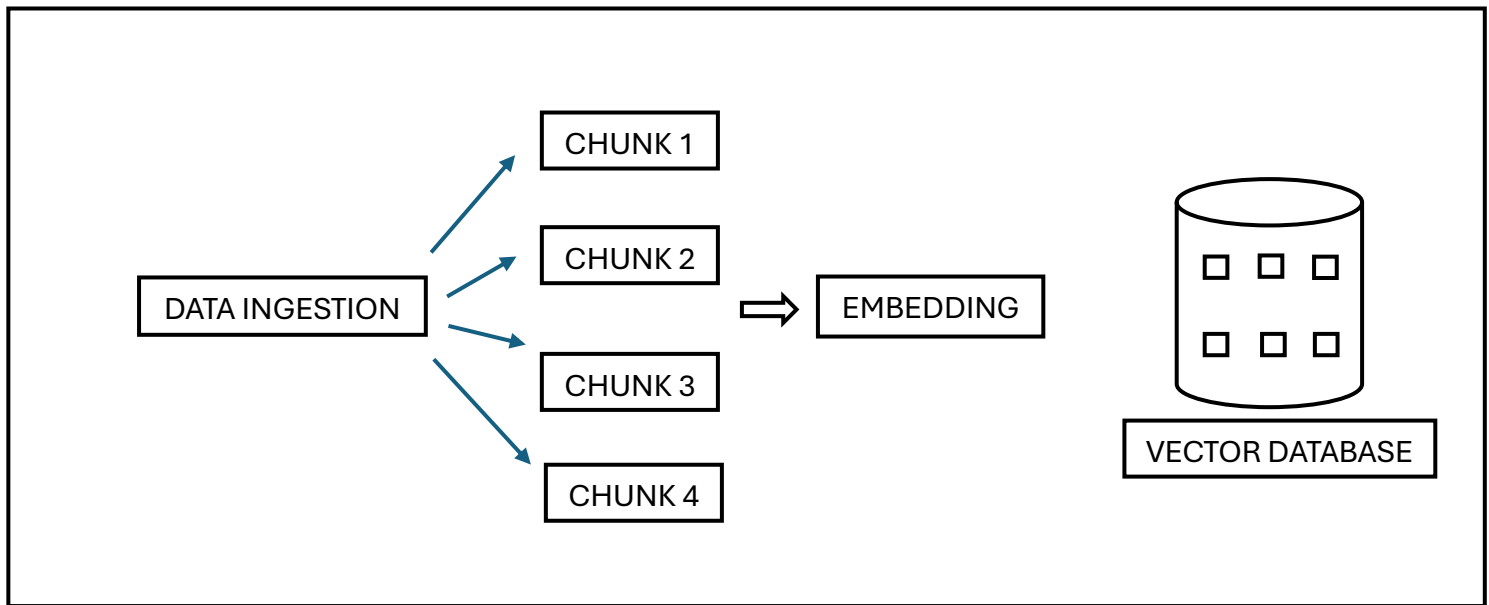
PERSON → QUERY + PROMPT → LLM → **Output**

A Complete RAG pipeline:



This diagram represents a complete Retrieval-Augmented Generation (RAG) system, which consists of two main parts - the Data Ingestion Pipeline and the Retrieval Pipeline. In the Data Ingestion Pipeline, data is first ingested and parsed to extract meaningful text. The processed data is then converted into embeddings and stored in a vector store (vector database). In the Retrieval Pipeline, when a user submits a query, the retriever searches the vector store to find the most relevant information. The retrieved context is then passed to the LLM model, which generates the final output. This architecture improves accuracy by allowing the LLM to use relevant external knowledge instead of relying only on its training data.

Data Ingestion Pipeline:



This diagram represents the Data Ingestion Pipeline in a RAG system. First, the data is ingested from sources like PDFs or documents. Then, the document is divided into smaller parts called chunks. Each chunk is converted into a numerical representation called an embedding. These embeddings are stored in a Vector Database, which allows efficient similarity search during the retrieval process. This pipeline prepares the data so that when a user asks a query, the system can quickly find the most relevant information.