**Retrieval augmented generation** or **RAG** allows large language models to handle a broader range of queries without the need of exponentially large training data sets.

**RAG** operates in a similar manner with the **retrieval** component gathering accurate and relevant information, like the architect. And the **generation** component creatively weaving this information into coherent and engaging responses similar to that of interior designer's role in home design.

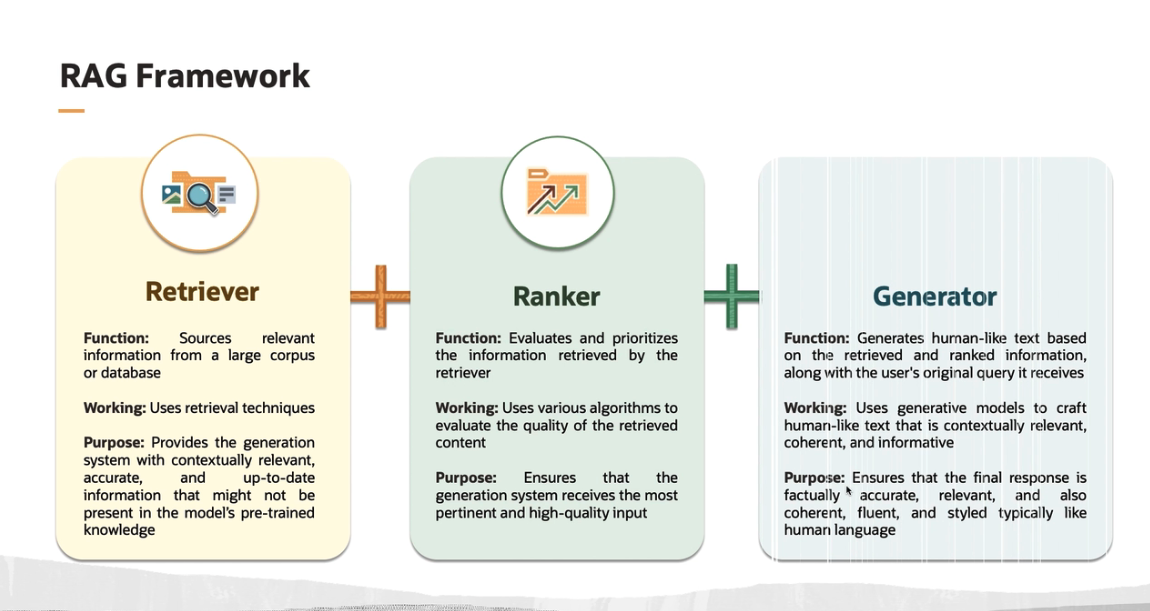
**Why we need RAG?**

- One of the limitations of standard LLMs is the reliance on the knowledge they were trained on. That knowledge might be outdated. Also, it falls short in tasks requiring a specific information. RAG address this by retrieving up-to-date information from external sources, thus enhancing the accuracy and relevance of the information it provides.

- RAG can significantly improve performance by retrieving documents or data that contain the exact information needed, something that pure generative models might struggle with.

- RAG models retrieves documents and pass them to a sequence to sequence model, such as encoder decoder architecture. A sequence to sequence model is particularly effective for tasks where the input and output sequences can have different lengths, such as machine translation, text summarization, and conversation modeling.

**RAG Framework:**

****

**1. Retrieval:** This component is responsible for sourcing relevant information from a large corpus or database. It acts like a search engine, scanning through vast amounts of data to find content that is pertinent(relevant) to the query at hand.

- It uses retrieval techniques, such as vector similarity search, keyword-based search, document retrieval or structured database queries to fetch data.

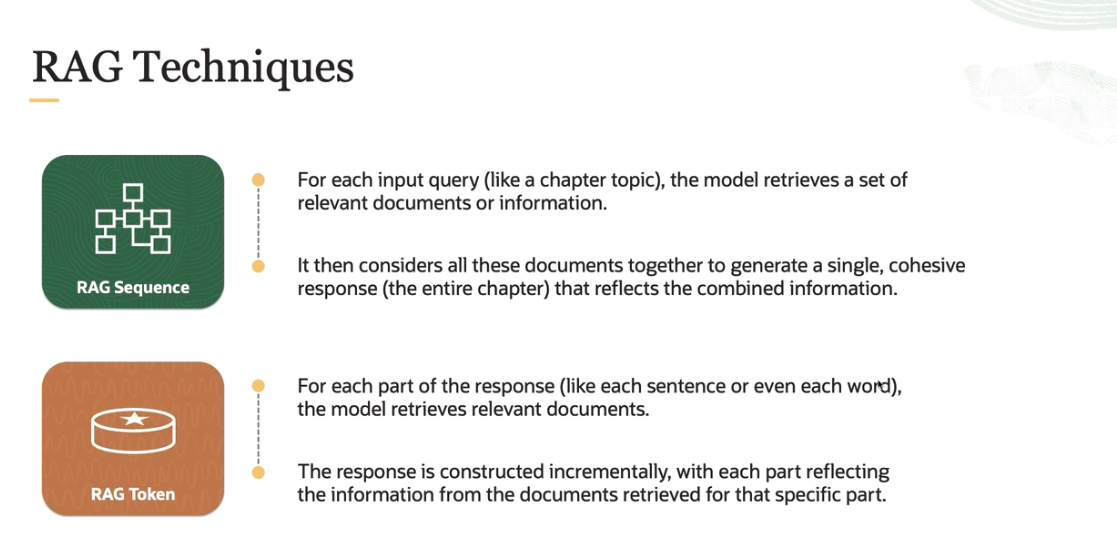
**2. Ranker:** The ranker's primary role is to evaluate and prioritize the information retrieved by the retrieval system. It sifts through the various pieces of data or documents that the retrieval system has gathered and ranks them based on their relevance and usefulness in answering the given query.

- By effectively ranking the retrieved information, the ranker ensures that the generation system receives the most pertinent and high quality input. This step is crucial for maintaining the accuracy and relevance of the responses generated by the model.

**3. Generator:** This is a language model, whose job is to generate human-like text based on the input it receives.

- It employs generative models to craft human-like text that is contextually relevant, coherent, and informative. It ensures that the final response is not only factually accurate and relevant but also coherent, fluent, and styled in a way that is typical of human language.

**RAG Techniques:**



**1. RAG Sequence model:** Think of RAG sequence model like a novelist who writes a book chapter by chapter. Similarly, for each input query, like a chapter topic, the model retrieves a set of relevant documents or information. It then considers all these documents together to generate a single cohesive response, that is the entire chapter that reflects the combined information.

Eg: Imagine you are writing a chapter about the French Revolution. You gather several books and articles on the topic, you read them all, and then write the entire chapter. This is how RAG sequence model works.

**2. RAG Token model:** RAG token model is like a journalist who writes an article, considering each piece of information or quote individually as he constructs his story. For each part of the response, like each sentence or even each word, the model retrieves relevant documents.

- The response is constructed incrementally with each part reflecting the information from the documents retrieved for that specific part. Imagine you are writing an article about a complex topic, such as climate change. For each point you make or each sentence you write, you look up specific articles or data to ensure that part of your story is accurate and relevant.

- Your final article is a combination of these individual pieces of researched information. And this is how RAG token works.

**Q. What are the key differences? And what is the scope of retrieval in those two cases?**

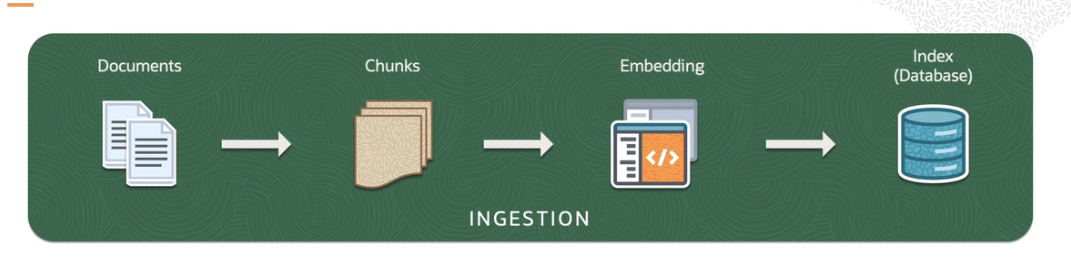
- The **RAG sequence** considers the entire input query at once for retrieval, while **RAG token** does this at more granular level, potentially leading to more varied and specific information integration.

- On the topic of response generation**, RAG sequence** is more about synthesizing a holistic response from a batch of information, whereas **RAG token** constructs the response in a more piecemeal fashion, considering different sources for different parts of response.

- Use **RAG token** when the task requires integrating highly specific and varied information from multiple sources into different parts of response, and Use **RAG sequence** when the task demands a more holistic and unified approach with responses that need to maintain thematic or contextual consistency across the entire text.

**RAG Pipeline (used in Natural Language Processing):**

The RAG architecture combines a retrieval-based component with a generative model to enhance the generation of text.



First phase is **Ingestion**, where documents are ingested into the system. Documents are nothing but the original data corpus, which can consist of a vast collection of text.

- The documents are then broken down into a smaller, more manageable piece, often referred to as **chunks**. This is typically done to improve the efficiency of processing and to focus on relevant sections of the text. Next, each chunk is then transformed into a mathematical representation called an **embedding**.

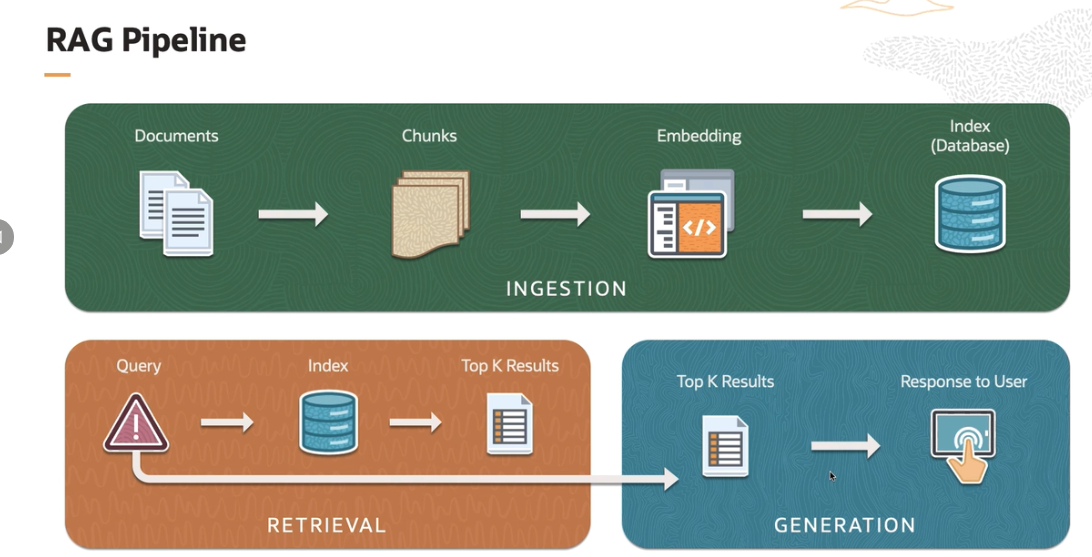
- These embeddings capture the semantic information of the text and allow for comparisons to be made in a numerical space. These embeddings are then indexed in a database that facilitates quick retrieval.

- The index is a data structure that allows the system to find and retrieve embeddings sufficiently when a query is made.

**In the next Phase**, the system uses the indexed data to find relevant information. Query is a user's input or question that need to be answered. The system uses this query to search through this indexed embeddings from the ingestion phase to find the most relevant chunks.

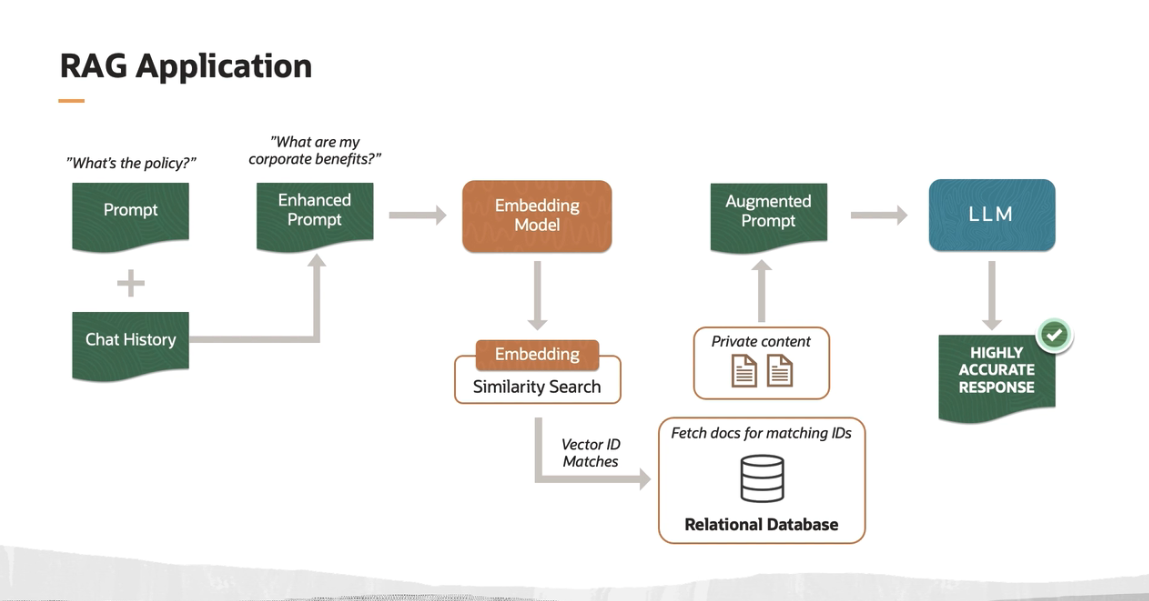
**From the retrieval process**, the system selects the top k most relevant results, which are the chunks that are most likely to contain information relevant to the query.

The **third and final phase** is **generation**. This is where the system generates a response based on the information retrieved. The selected chunks from the retrieval phase are fed into the generative model. The generative model, often a neural network like a transformer, uses the context provided by this top k chunks to generate a coherent and contextually relevant response to the query.



This approach is particularly useful in scenarios where generative models need to be supplemented with specific information that may not be present in their training data.

**Q. Now delve into how this process unfolds.So imagine an employee asking a simple question. What is the policy?**



- The question is clear. But for a language model to provide a relevant answer, it needs context. And this is where the RAG model starts to shine. So in addition to this initial prompt, the model considers the entire chat history.

- This allows the model to understand the conversation's context, tailoring its response to be more accurate and specific. The prompt and chat history are then combined to form what we call an enhanced prompt. Think of it as a more informed question that provides a clearer picture for the language model to understand.

- The enhanced prompt is then passed through a embedding model, which transforms the text into a mathematical vector. These vectors are representations that capture the semantic meaning of the prompt in a high-dimensional space.

- Next, we perform an embedding similarity search, Here, the vector of our enhanced prompt is compared against a database of other vectors. And this is how the model finds the most relevant information to this query.

- The search results in vector ID matches, which are essentially references to documents that have the closest semantic similarity to our query.

- In many cases, these documents contain private content, such as corporate policies or specific employee benefits information stored securely in a relational database. The system then retrieves the document associated with the matching vector ID. This process ensures that the response is both accurate and customized to the user's need.

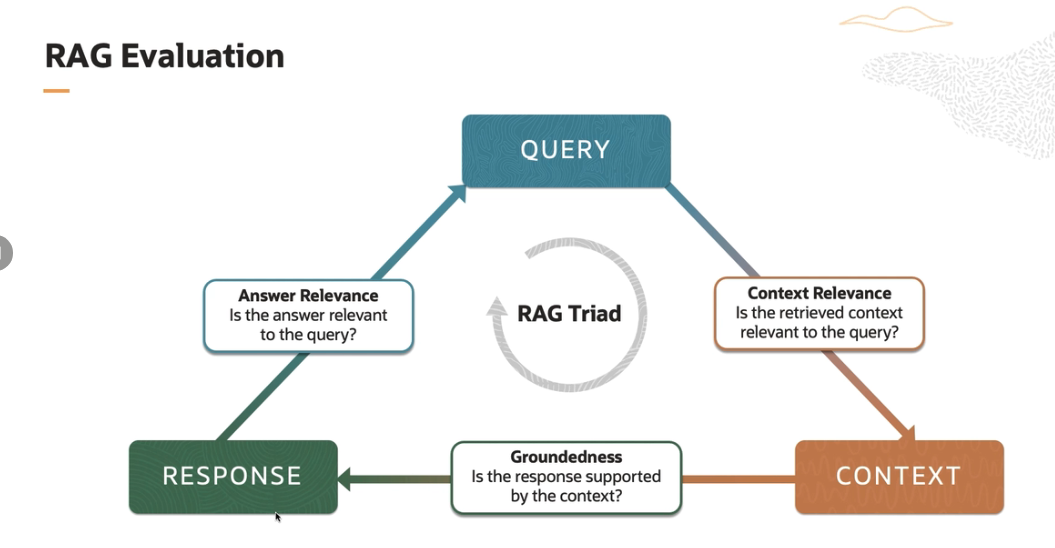
- So once we have the relevant documents, they are used to augment the initial prompt. This step ensures that the model's response is not only relevant but also contains validated information. This augmented prompt is then fed into a large language model, which generates the response.

Even though RAG have the access to the most updated database and gives accurate answers, it does not fully eliminate the risk of hallucinations for various reasons.

- First, the retrieval could simply fail to retrieve sufficient context or get the relevant one.

- Second, the response generated by a RAG application was not supported by the retrieved context, but would have been mostly influenced by the LLM and its training data.

- And finally, a RAG application may retrieve relevant pieces of context, then leverage them to produce a grounded response and yet still fail to address a user query. For these, RAG triad comes handy.



1. So the first one in RAG triad is **context relevance**: So context relevance refers to how well the RAG responses are aligned with the context of the conversation. This includes understanding the ongoing dialogue, the user's intent, and any background information or conversational history. A RAG application with high context relevance can maintain a coherent and meaningful conversation over multiple turns.

2. The second one is **groundedness**. It indicates the chatbot's ability to provide responses that are not only plausible but also accurate and based on reliable information. t can be evaluated by comparing the RAG responses to trusted sources of the information or through expert assessment, especially for domain specific applications.

3. The third part of the RAG triad is **answer relevance**. So the answer relevance refers to the degree to which the RAG responses directly address the user's queries. It's about providing responses that are not only contextually appropriate, but also specifically answer or relate to the user's questions or statement.