Summary and Highlights: Advanced Retrievers for RAG

Congratulations! You have completed this lesson. At this point in the course, you know:

- A LangChain retriever is an interface that returns documents based on an unstructured query
- There are several different types of LangChain retrievers
- The vector store-based retriever retrieves documents from a vector database
- A vector store-based retriever can be created directly from the vector store object with the retriever method by using similarity search or MMR
- That similarity search is when the retriever accepts a query and retrieves the most similar data
- MMR is a technique used to balance the relevance and diversity of retrieved results
- The multi-query retriever uses an LLM to create different versions of the query, generating a richer set of retrieved documents
- The self query retriever converts the query into two components, a string to look up semantically, and a metadata filter to accompany it
- The parent document retriever has two text splitters: a parent splitter that splits the text into large chunks to be retrieved, and a child splitter that splits the document into small chunks to generate meaningful embeddings
- The core LlamaIndex index types are the VectorStoreIndex, the DocumentSummaryIndex, and the KeywordTableIndex
- The VectorStoreIndex stores vector embeddings for each document chunk, is best suited for semantic retrieval, and is commonly used in pipelines that involve large language models
- The DocumentSummaryIndex generates and stores summaries of documents, which are used to filter documents before retrieving the full content, and is useful when working with large and diverse document sets
- The KeywordTableIndex extracts keywords from documents and maps them to specific content chunks, and is useful in hybrid or rule-based search scenarios
- The Vector Index Retriever uses vector embeddings to find semantically related content, and is ideal for general-purpose search and RAG pipelines
- The BM25 Retriever is a keyword-based method for ranking documents, and it retrieves content based on exact keyword matches rather than semantic similarity