Retrieval Augmented Generation (RAG)

By RDP Team

REVOLUTION DATA PLATFORMS

Microsoft Solutions Partner

Data & AI

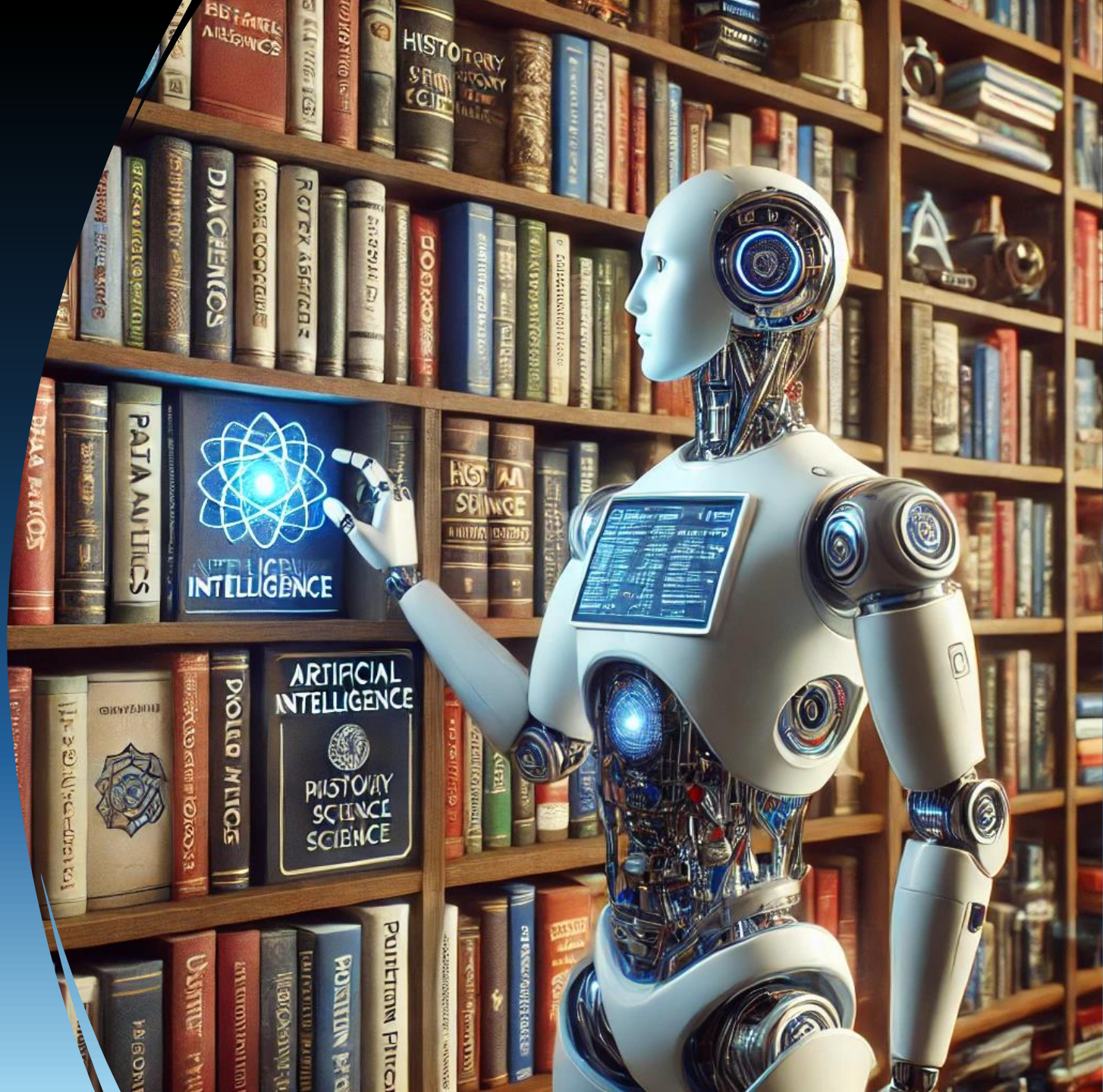
Azure

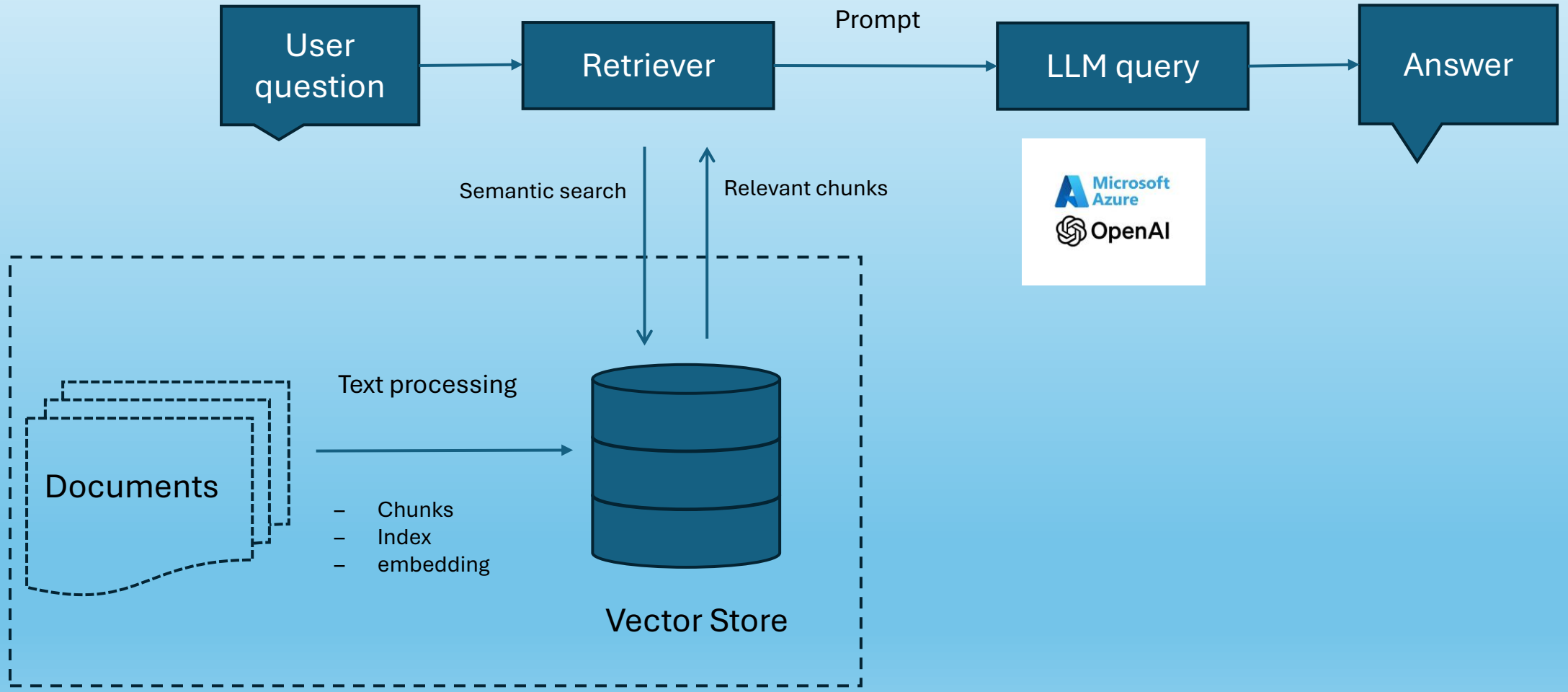
How RAG works :

- RAG combines two key technologies :

➢ **Retrieval** (finding relevant information)

➢ **generation** (creating responses)




User question
Retriever
Prompt
LLM query
Answer
Semantic search
Relevant chunks
Microsoft Azure
OpenAI
Text processing
Documents
– Chunks
– Index
– embedding
Vector Store

Retrieval

Document Pre-processing

Retrieval Engine

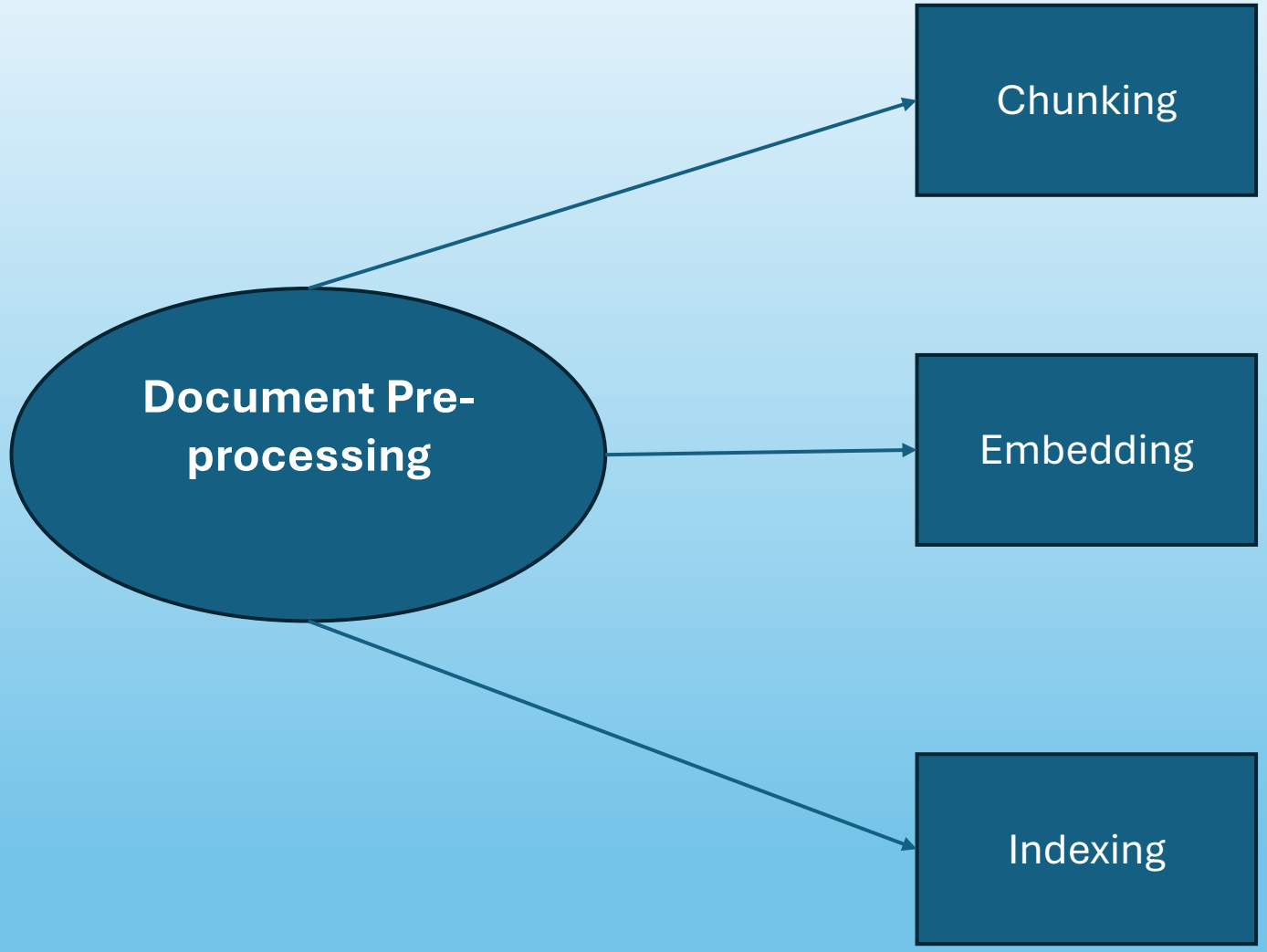
Vector Store

Generation

Language Model

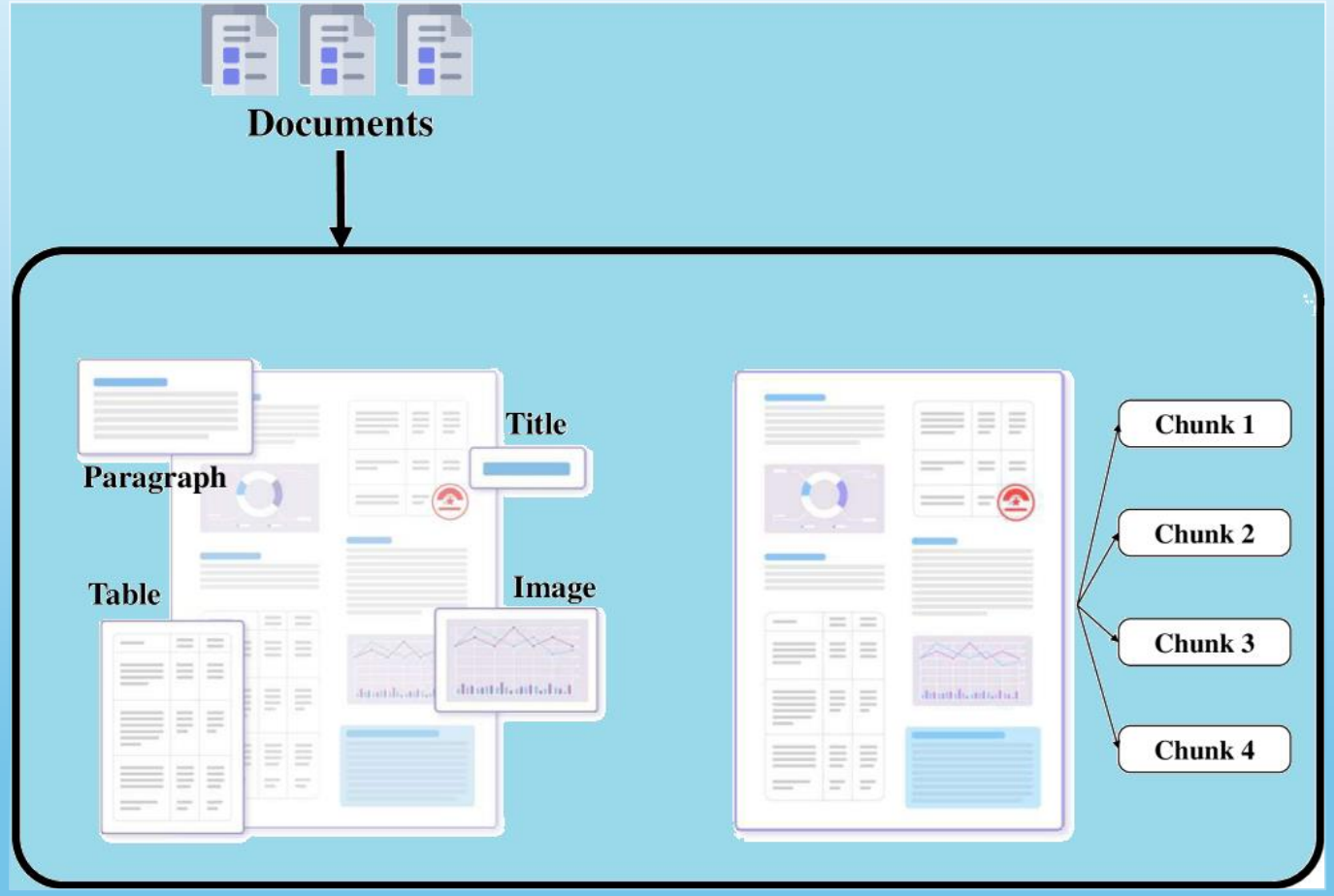
Context integration

Response Formulation


Document Pre-processing
Chunking
Embedding
Indexing

Chunking

- process of breaking large documents into smaller, manageable sections or "chunks
- Allows to search through relevant parts of the document, rather than processing the entire document.
- Improves retrieval efficiency and accuracy by focusing on smaller, more precise sections of text.
- Ensures that the retrieval engine extracts the most relevant information from large documents.



Fixed-size Chunking:

•Documents are split into fixed-size blocks, such as 512 tokens or words, regardless of the content.

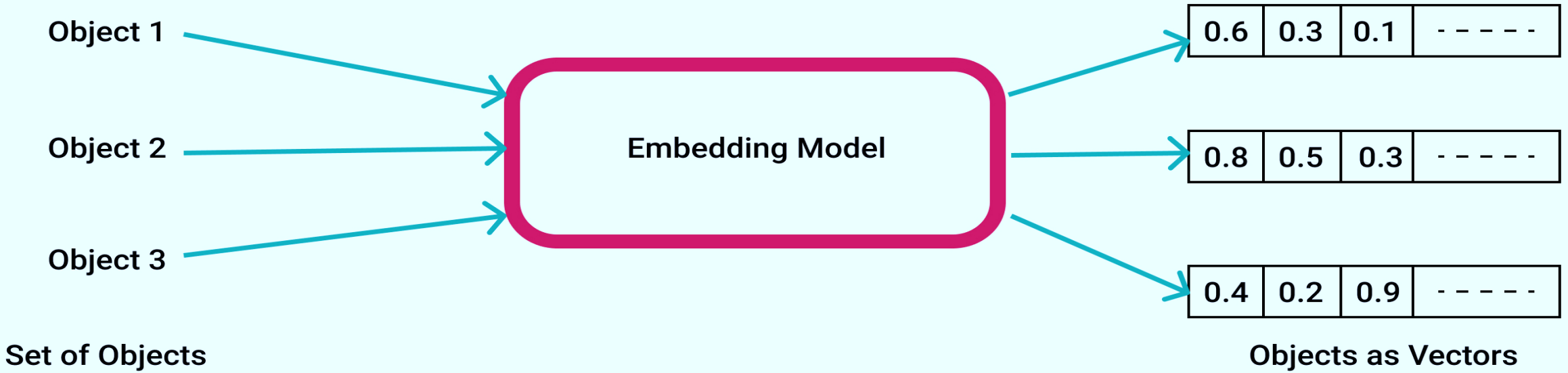
•**Pros:** Easy to implement.
•**Cons:** May split important context between chunks.

Semantic Chunking:

•Chunks are divided based on natural language boundaries, such as sentences, paragraphs, or topics

•**Pros:** Preserves context, better suited for retrieval.
•**Cons:** More complex to implement.

Embedding and Vector representation



- Enable the system to perform **semantic search**
- Allow the system to compare the similarity of different pieces of text.

process of converting text into high-dimensional vectors (numeric representations)

Common Embedding methods

text-embedding-ada-002

•developed by OpenAI and hosted on Azure
•It uses deep learning techniques
•capture complex semantic relationships.
•High-dimensional embeddings
•performs well on a wide range of NLP tasks

Use Cases

•Natural Language Processing tasks like text classification.
•Semantic search and information retrieval.
•Machine translation and sentiment analysis.

TF-IDF (Term Frequency-Inverse Document Frequency) Model:

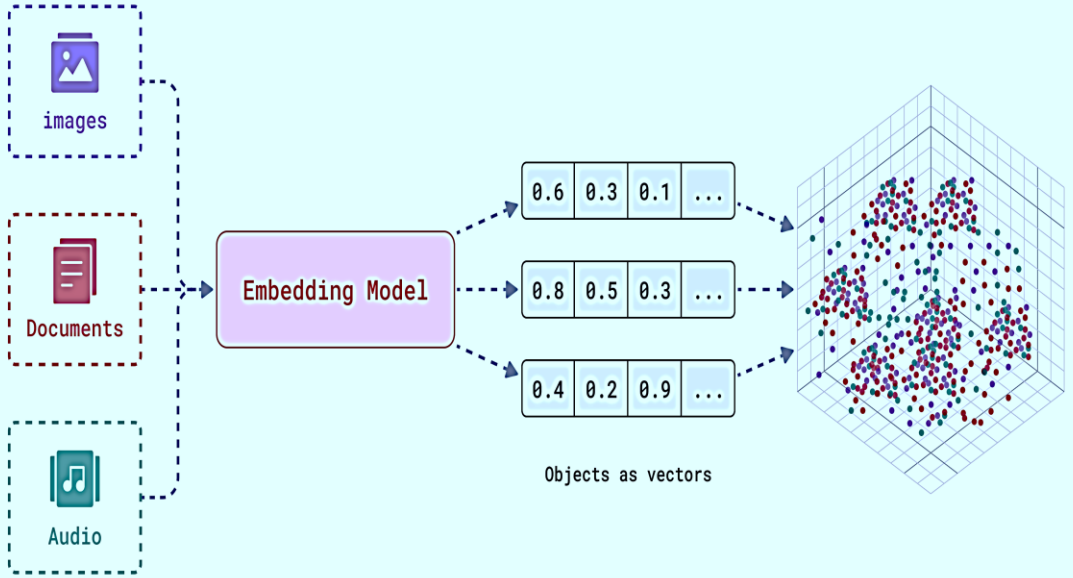
- evaluates the importance of a word in a document

•Lower-dimensional embeddings (dimension depends on vocabulary size, 27 in this case).
•Based purely on word frequency and uniqueness, not on semantic meaning.
•Simple and computationally efficient.

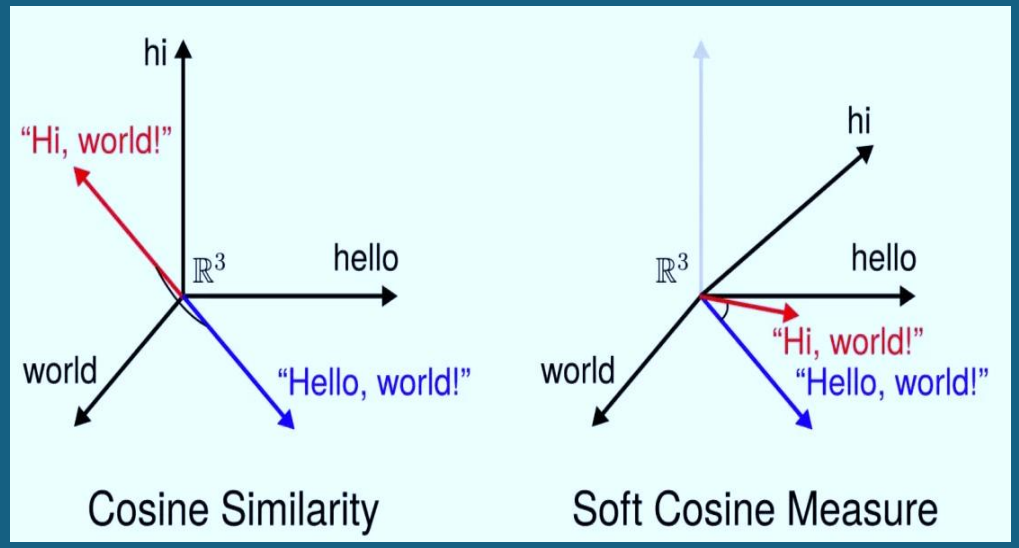
Method	Pros	Cons
TF-IDF	Simple, interpretable	Doesn't capture context, large vectors
Bag of Words	Simple, fast	Ignores grammar and word order
Word2Vec	Captures semantic meaning	Requires large corpus to train well
GloVe	Captures both local and global context	Requires memory for co-occurrence matrix
FastText	Works well with rare words	Slightly more complex than Word2Vec
Doc2Vec	Useful for document-level tasks	More complex to train and fine-tune
LSA	Reduces dimensionality, finds topics	Needs tuning of SVD, may lose some interpretability
BERT	Context-aware embeddings	Computationally intensive

Vector Representation

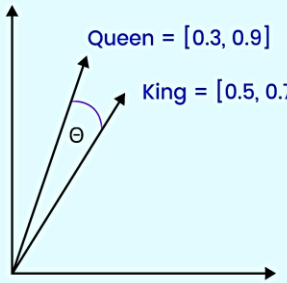
- **High-dimensional Vectors:**
- Each dimension captures a feature of the data



- **Similarity Measures:** Cosine similarity



$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$



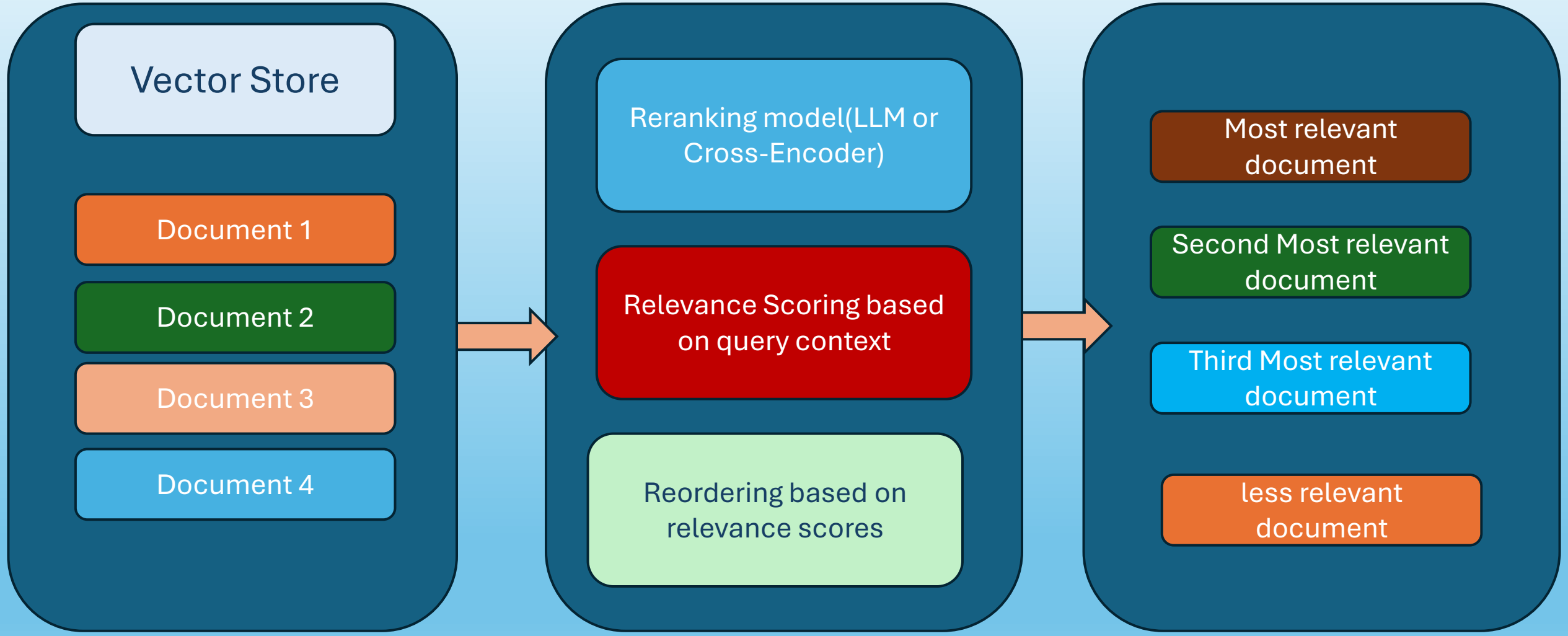
$$\text{Cos(Queen, King)} = \frac{(0.3*0.5)+(0.9*0.7)}{\sqrt{0.3^2+0.9^2} * \sqrt{0.5^2+0.7^2}}$$

$$= \frac{0.15+0.63}{\sqrt{0.9^2} * \sqrt{0.74}}$$

$$= \frac{0.78}{\sqrt{0.666}}$$

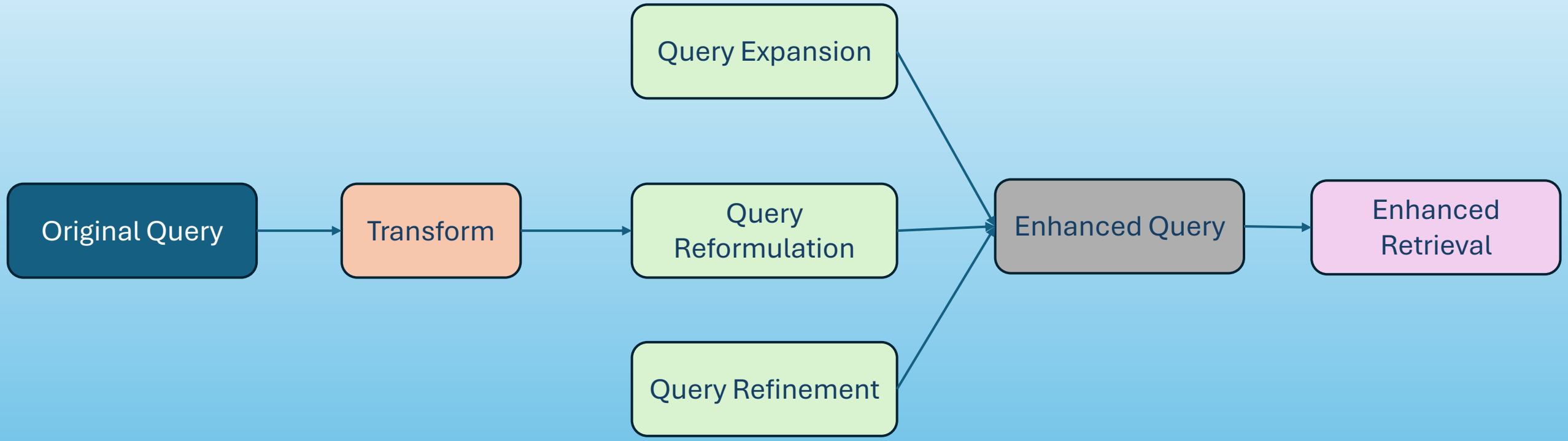
$$= 0.03$$

Reranking


Vector Store
Document 1
Document 2
Document 3
Document 4
Reranking model(LLM or Cross-Encoder)
Relevance Scoring based on query context
Reordering based on relevance scores
Most relevant document
Second Most relevant document
Third Most relevant document
less relevant document

Method	Description	Pros	Cons
Semantic Similarity	Reranks based on semantic closeness to the query	-Captures meaning beyond -Works well with embeddings	Computationally intensive May miss syntactic importance
Learning to Rank	Uses machine learning to optimize ranking	Can incorporate multiple features Adaptable to specific domains Improves with more data	-Requires training data - Can be complex to implement May overfit on training data
Cross-Encoder	Uses transformer models to compare query-document pairs	High accuracy Captures complex relationships Can be fine-tuned	Slower than other methods Resource-intensive
Hybrid Methods	Combines multiple reranking approaches	Leverages strengths of different methods Can be optimized for specific use cases	Increased complexity May require careful tuning Potentially higher latency

Query Transformation



Query Expansion:

- Adds related terms or synonyms to the original query.
- Good for improving recall and handling vocabulary mismatches

Example :

car repair

Expanded query:
"car repair OR auto mechanic OR vehicle maintenance OR garage service"

Query Reformulation:

- Rewrites the query to improve its effectiveness
- Good clarifying intent and handling complex queries

Example :

How tall is the tower in Paris?

Reformulated query:
What is the height of the Eiffel Tower?

Query Refinement:

- Narrows down or specifies the query based on context or user feedback
- for improving precision and personalizing results

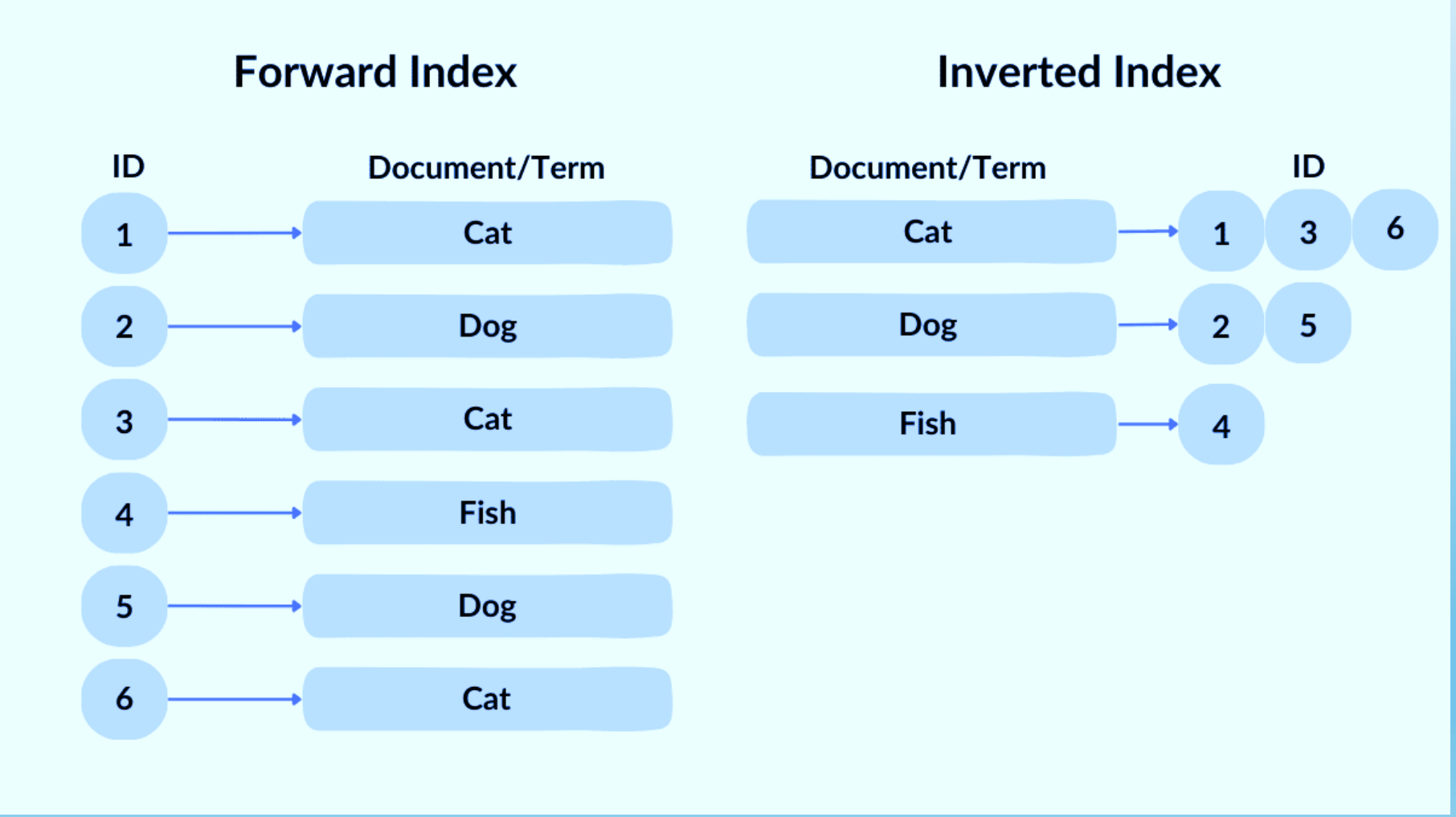
- Example:

best laptop

Refined query:
"best laptop for gaming under $1000 with at least 16GB RAM"

Indexing

searchable structure (like a database) where documents are mapped to keywords or embeddings

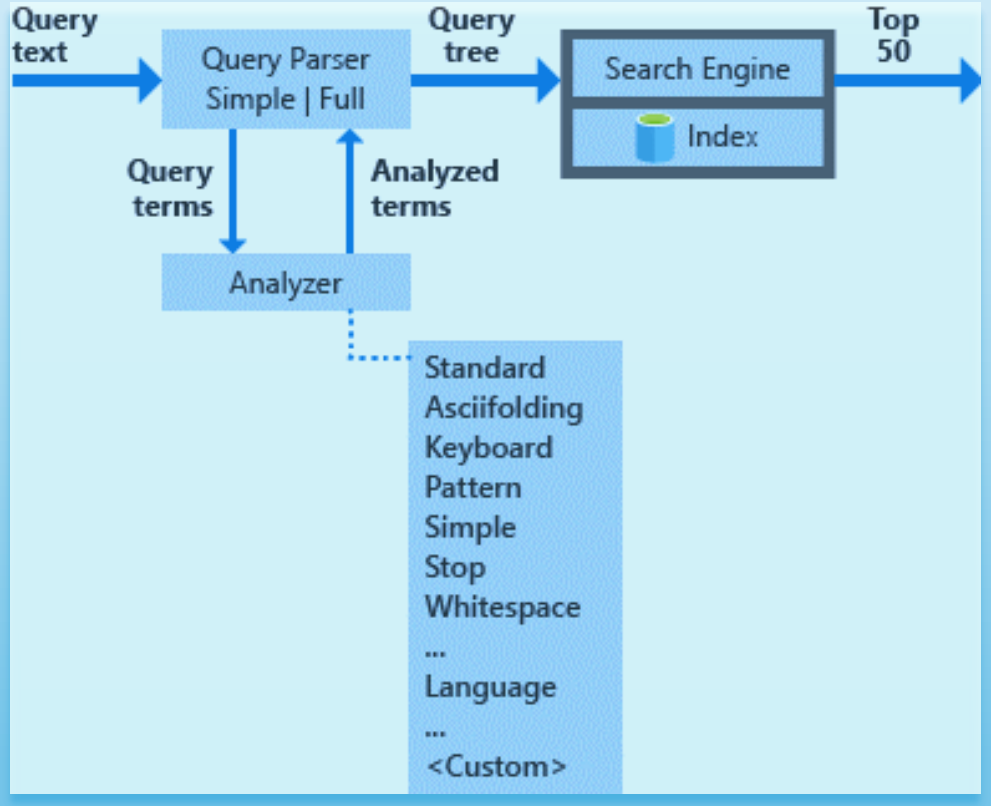


Azure AI Search

Overview
•Cloud-based search-as-a-service by Microsoft.
•Incorporates AI capabilities into search solutions.

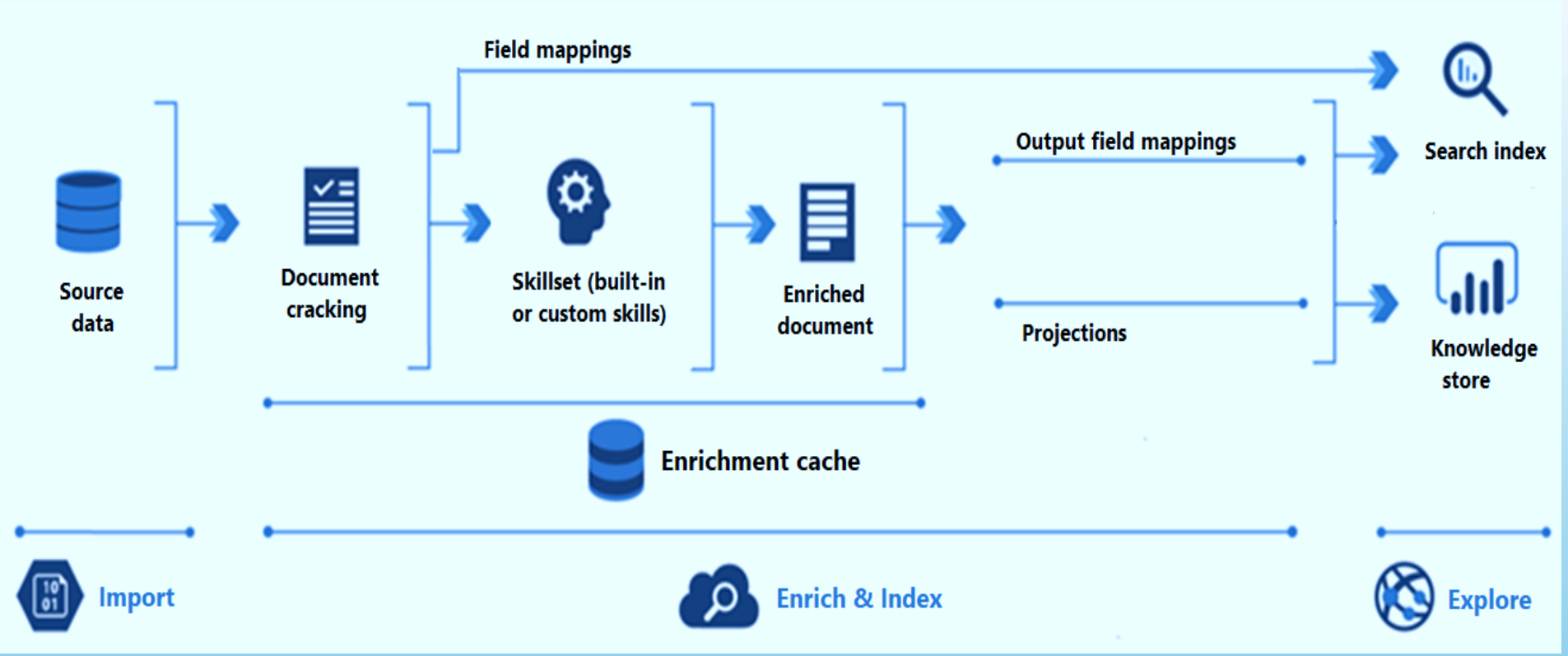
- **Full-Text Search**
- **AI Enrichment**
- **Semantic Search**
- **Vector Search**

Full-Text Search



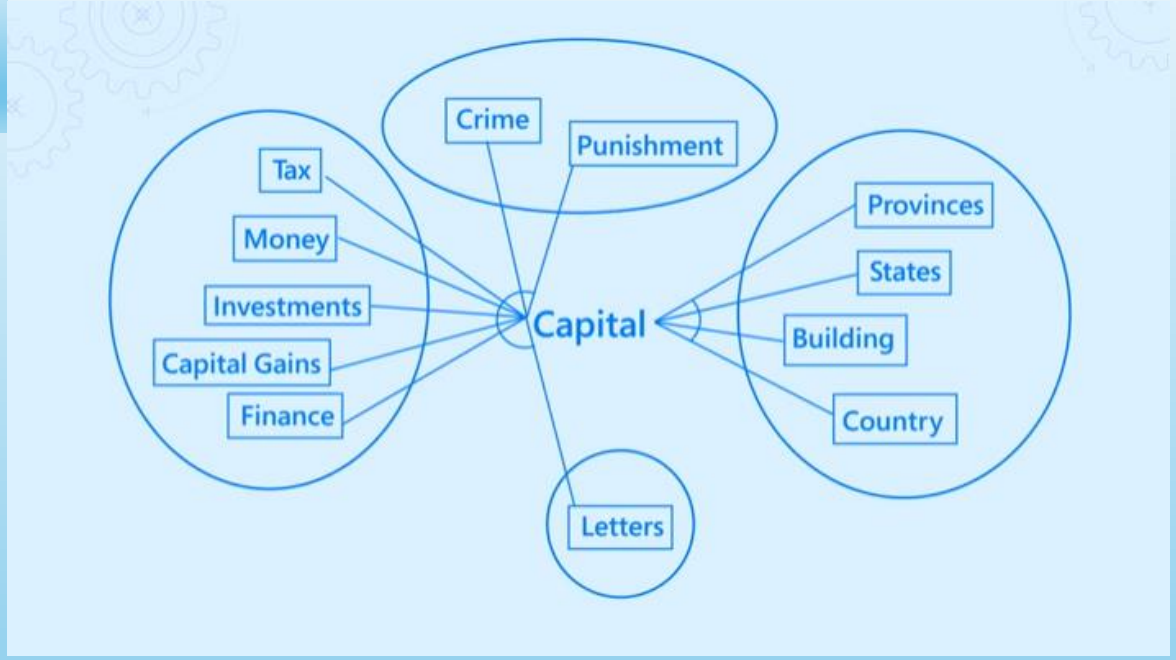
Key components	Functional description
Query parsers	Separate query terms from query operators and create the query structure (a query tree) to be sent to the search engine.
Analyzers	Perform lexical analysis on query terms. This process can involve transforming, removing, or expanding of query terms.
Index	An efficient data structure used to store and organize searchable terms extracted from indexed documents.
Search engine	Retrieves and scores matching documents based on the contents of the inverted index.

AI Enrichment



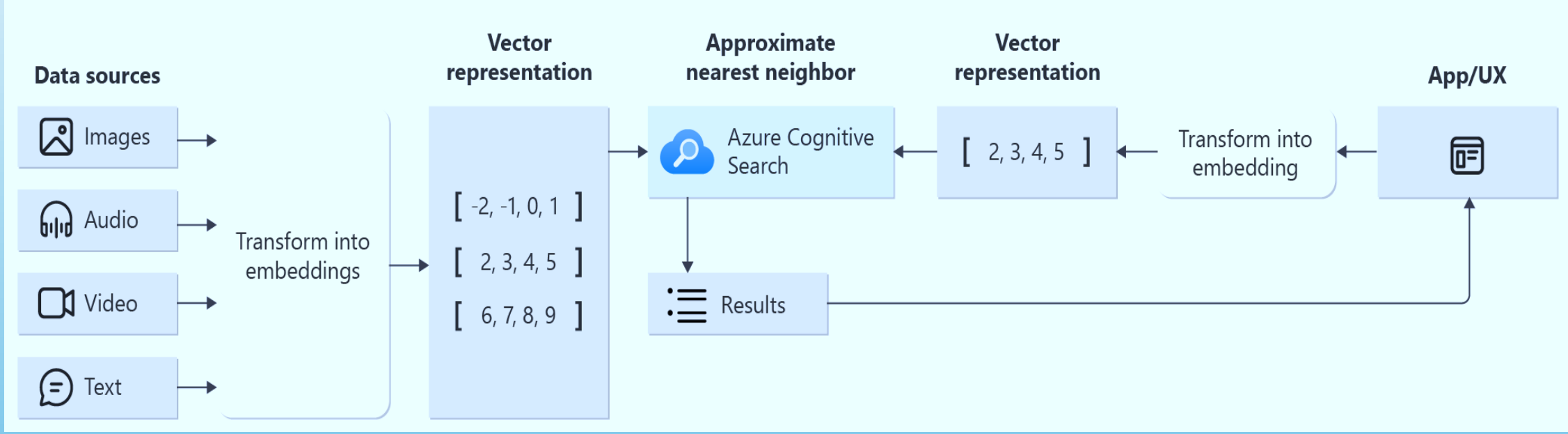
- **Image to Text**: Extract text from images or scanned documents (OCR) for indexing.
- **Entity Recognition**: Identify key entities like people, places, and organizations in unstructured text.
- **Sentiment Analysis**: Analyze customer feedback data for sentiment during the indexing process

Semantic Search



Feature	Description
Semantic ranking	Uses the context or semantic meaning of a query to compute a new relevance score over preranked results.
Semantic captions and highlits	Extracts key sentences and phrases with highlights for easy scanning, helping elevate relevant terms for dense content.
Semantic answers	Provides a direct answer to question-like queries by returning text with answer characteristics.

Vector Search



- **Recommendation Systems**: Find similar products based on user search patterns or embeddings.
- **Multimedia Search**: Retrieve similar images or videos based on vector embeddings.
- **Document Retrieval**: Find contextually similar documents using embeddings from text.


Thank you
AI