RAG Made Simple: Embeddings, Indexing, Naive RAG & HyDE

1. What Are Embeddings?

Embeddings convert sentences into numbers (vectors) that represent **meaning**. Think of it as giving each idea a **GPS coordinate** on a *map of meanings*. Similar ideas are stored close together, while unrelated ideas are far apart.

Example: 'Open savings account' → [0.12, 0.53, ...] | 'Create a bank account' → [0.11, 0.54, ...] (close together). 'Pizza recipe' → [0.88, -0.23, ...] (far away).

2. What Is Indexing?

After embeddings are generated, they are stored in a **vector database** like Chroma. Indexing acts like a **library catalog** — it quickly points to the right shelf of information, making search faster and semantic (based on meaning, not just exact keywords).

3. Naive RAG Workflow

1. User enters a query → 2. Convert query into an embedding → 3. Retrieve closest documents → 4. Combine query + documents → 5. Send to LLM → 6. Generate answer.

Key Point: Uses a **frozen LLM**. No retraining needed — we only inject context.

4. HyDE RAG Workflow

HyDE improves Naive RAG by letting the LLM first generate a **hypothetical answer**. This imagined answer is embedded and used to fetch better matching documents. The result: **higher recall** and **more accurate context-rich answers**.

5. Naive RAG vs HyDE RAG

Feature	Naive RAG	HyDE RAG
Embeds query directly	■	■
Embeds hypothetical answer	■	■
Handles vague queries	Weak	Strong
Retrieval accuracy	Good	Better

6. Real-Life Analogy

Naive RAG is like asking a friend for book recommendations using **exact words**. HyDE is smarter — your friend imagines what you *really want* and finds better answers.

7. Quick Q&A;

Q: Why embeddings instead of keywords?
A: Embeddings search by **meaning**, not exact matches.

Q: Why HyDE over Naive RAG?
A: HyDE retrieves better documents, especially for vague queries.

Q: Do embeddings store raw text?
A: No — they only store numerical meaning, making them **privacy-safe**.