Demo Use-Case 2- Hello RAG in SQLite OpenAI

This project will walk you through building a **Retrieval-Augmented Generation** (RAG) workflow using SQLite with OpenAI LLM + Embeddings + SQL + Python.

We will go from data setup to an AI-powered query answering system ·

Python required (OpenAI API).

Project Goal

Build an intelligent **Question Answering System** that:

- Stores internal documents (policies, FAQs, product info, etc.)
- Embeds them using OpenAI embedding models
- Finds the most relevant documents for a user's query
- Uses OpenAI chat completion to generate a natural-language answer

SQLite + Python.

Architecture Overview

Step	Component	SQLite/OpenAI Used
1	Data ingestion	CREATE TABLE, INSERT
2	Vector embedding creation	OpenAI embeddings

Step	Component	SQLite/OpenAI Used
3	Semantic similarity search	NumPy cosine similarity
4	Answer generation	OpenAI chat completion
5	Optional summarization	OpenAI summarize

Step 1: Create Sample Knowledge Base

```
CREATE TABLE COMPANY_DOCS AS
SELECT 1 AS DOC_ID, 'Employees are allowed to work remotely up to two days per week.' AS CONTENT UNION ALL
SELECT 2, 'Annual leave requests should be submitted at least two weeks in advance.' UNION ALL
SELECT 3, 'The company provides full health insurance coverage to all permanent employees.' UNION ALL
SELECT 4, 'Employees must complete mandatory cybersecurity training every year.' UNION ALL
SELECT 5, 'Performance reviews are conducted twice a year: mid-year and end-of-year.';
```

You now have a basic knowledge base with five HR/policy documents.

Step 2: Create Embeddings for All Documents

```
CREATE TABLE COMPANY_DOCS_EMBEDDINGS AS
SELECT
  DOC_ID,
  CONTENT,
OPENAI_EMBED(CONTENT) AS EMBEDDING
FROM COMPANY_DOCS;
```

This creates 1536-dimensional embeddings for each document.

Run this to confirm:

```
SELECT * FROM COMPANY_DOCS_EMBEDDINGS LIMIT 2;
```

Step 3: Run Semantic Search (Vector Similarity)

Now test with a user query, for example:

“Does the company allow remote work?”

```
WITH QUERY_EMB AS (
SELECT OPENAI_EMBED('Does the company allow
remote work?') AS VECTOR
)
SELECT
  d.DOC_ID,
COSINE_SIMILARITY(d.EMBEDDING, q.VECTOR) AS SIMILARITY,
  d.CONTENT
FROM COMPANY_DOCS_EMBEDDINGS d, QUERY_EMB q
ORDER BY SIMILARITY DESC
LIMIT 3;
```

You should see the remote work policy document (DOC_ID = 1) ranked at the top.

Step 4: Generate an Answer using OpenAI

Now we'll feed the top-matched documents into an **LLM prompt** for final answer generation.

```
WITH QUERY AS (
  SELECT 'Does the company allow remote work?' AS QUESTION
),
RELEVANT_DOCS AS (
  SELECT
    d.CONTENT
  FROM COMPANY_DOCS_EMBEDDINGS d, QUERY q,
      LATERAL (
SELECT OPENAI_EMBED(q.QUESTION) AS
VECTOR
      ) qe
  ORDER BY VECTOR_COSINE_SIMILARITY(d.EMBEDDING, qe.VECTOR) DESC
  LIMIT 3
),
CONTEXT AS (
  SELECT LISTAGG(CONTENT, '\n') AS DOC_CONTEXT FROM RELEVANT_DOCS
)
SELECT OPENAI_COMPLETE(
'gpt-4o-mini',
  CONCAT(
```

```
  'You are an HR assistant. Use the following context to answer the question clearly.\n\n',
  'Context:\n', (SELECT DOC_CONTEXT FROM CONTEXT), '\n\n',
  'Question: ', (SELECT QUESTION FROM QUERY), '\nAnswer:'
 )
) AS GENERATED_ANSWER;
```

Expected Output:

The company allows employees to work remotely up to two days per week.

Step 5: Optional — Add Summarization or Sentiment

You can summarize long retrieved docs before passing to the LLM:

```
SELECT OPENAI_SUMMARIZE(CONTENT) AS SHORT_SUMMARY
FROM COMPANY_DOCS
WHERE DOC_ID = 1;
```

Or analyze tone:

```
SELECT OPENAI_SENTIMENT('I really enjoy the flexible work policy!') AS
SENTIMENT;
```

Step 6: Validate RAG Workflow

Step	Action	Function Used	Example
1	Ingest documents	CREATE TABLE	HR policies
2	Embed documents	EMBED_TEXT_768	Converts text → vectors

Step	Action	Function Used	Example
3	Retrieve relevant docs	NumPy cosine similarity Finds top-K context	
4	Generate final answer OpenAI		LLM answers user query
5	(Optional) Summarize	SUMMARIZE	Simplify context

Real-World Use Cases

Use Case	Description	SQLite/OpenAI Function
HR Knowledge Bot	Employees ask HR policy questions	Chat + embeddings
Customer Support Assistant	Summarize FAQs and return answers	Summarize + embeddings
Product Catalog Search	Semantic search for product info	NumPy cosine similarity
Document Summarization	Short summaries for legal docs	Summarize
Internal AI Agent	Combine multiple OpenAI functions	Chat, sentiment, TRANSLATE, etc.

Key Functions Used

Function	Purpose
OpenAI embeddings	Convert text to vector embeddings
NumPy cosine similarity Compute similarity	
Chat completions	Generate text / answers
Summarize	Create short summaries
Translate prompt	Translate content
Sentiment prompt	Analyze tone of text

Final Output

After completing all steps, you will have a working local RAG pipeline:

- No external vector database
- API key required
- Portable, auditable, and runnable on a laptop