Let's Build an Advanced RAG combing RAG and Text-to-SQL

—------

Before we begin, here's a quick demo of what we're building!

We will be using

@OpenAI: OpenAI model for language processing

@llama_index: LlamaCloud index for RAG-based retrieval

Demo: https://github.com/afizs/ai-engineering-hub/blob/main/llamacloud_sql_router/demo.mp4

—

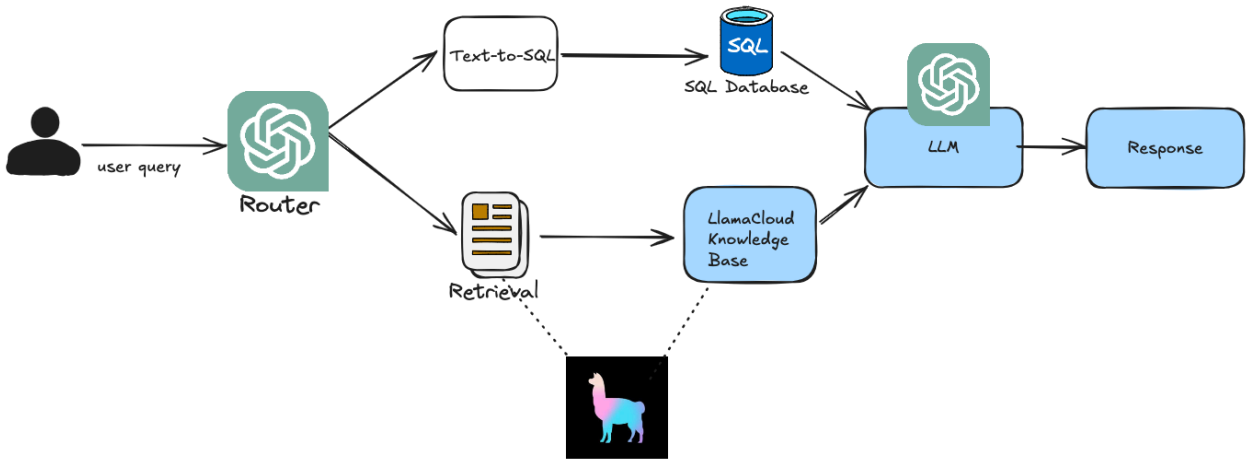
How it works:

1. User inputs a query (e.g., "Which city has the highest population?")

2. AI selects the appropriate tool (SQL for structured queries, Llama Cloud for semantic ones)

3. The tool executes the query and retrieves the response.

4. Results are returned to the user



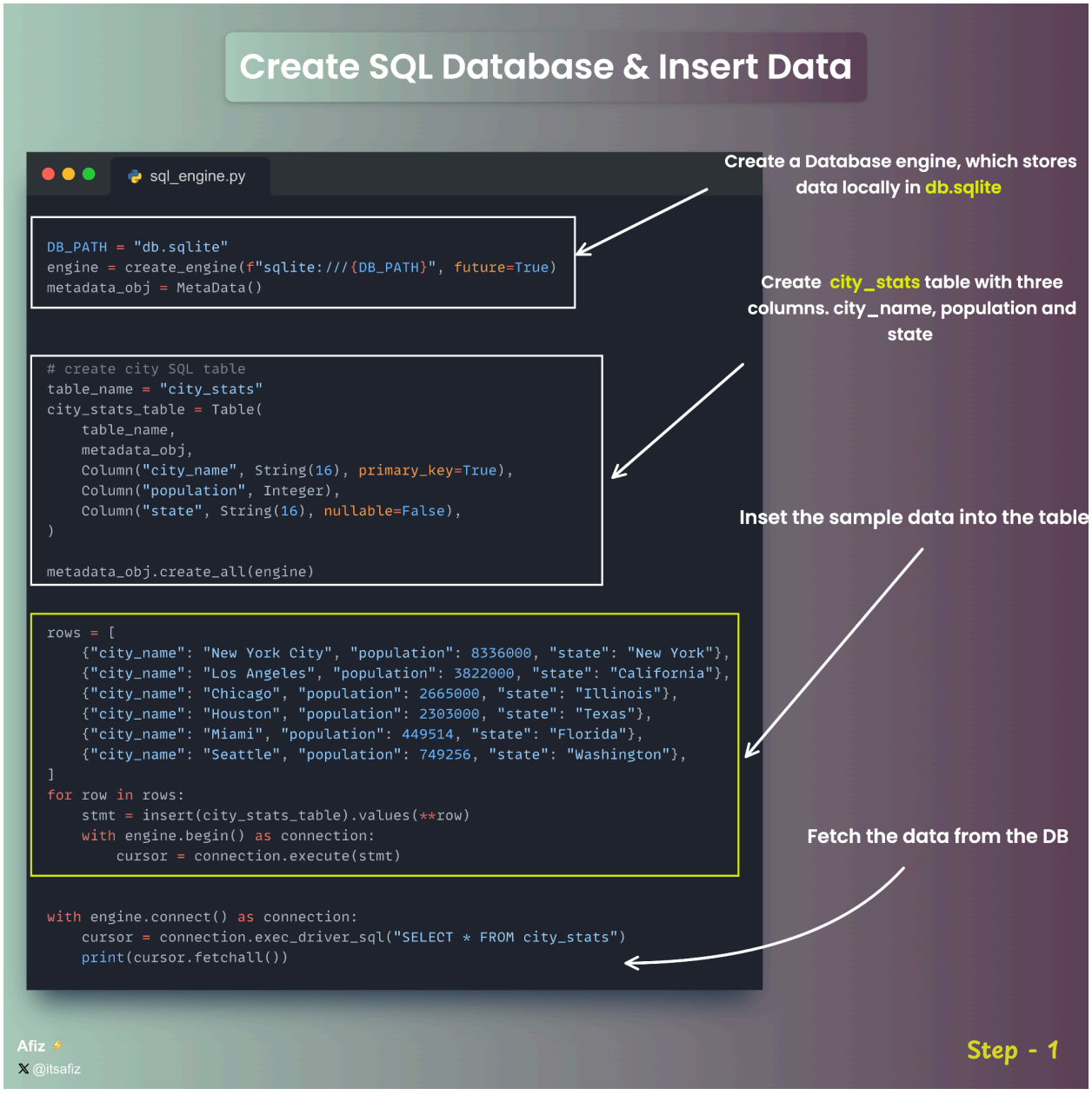
—

0. Setup your API keys and project and index name of Llamaindex

1. Create SQL Database and Insert Data

We will use SQLite for creating the Database. This is a 3 step process.

- Create the SQL Engine
- Create table and with three columns
- Insert the sample data



2. AI powered SQL Query Engine

   OpenAI gpt-3.5-tubo model is used for natural language processing.

3. Setup LlamaIndex Query Engine for RAG

   Checkout the Github repo for the steps to create index on LlamaIndex Cloud

4. Create Query Engine Tools
    - **`sql_tool`** – Uses `sql_query_engine` to retrieve city population and state data via **SQL queries**.
    - **`llama_cloud_tool`** – Uses `llama_cloud_query_engine` to answer **semantic questions** about US cities.

5. Let’s create a workflow that acts as an agent around the two query engines.

First in the workflow, we need 4 events as follows

5.1 Define the `RouterOutputAgentWorkflow` Class

This class **manages AI tool selection and execution**

It initializes:

- `tools`: A list of available **query tools** (SQL, Llama Cloud)
- `chat_history`: Stores past **user queries and responses**
- `llm`: Uses **GPT-3.5-turbo** as the default language model

6. Define Workflow Steps for Processing Messages and Calling Tools

This workflow consists of 5 steps.

1. Process User Message

6.2 Choose an AI Tool

GPT-3.5-turbo model analyzes the user message. It selects the most relevant tool (`sql_tool` or `llama_cloud_tool`).

6.3. Dispatch Tool Calls

Sends the tool request to the appropriate AI tool.

6.4 Execute the Tool Call

- Calls the selected tool (`SQL tool` or `Llama Cloud`).
- Executes the query and stores the result in a chat message.

6.5 Gather and Return the Tool Results

- Retrieves all responses from the AI tool.
- Adds the response to chat history.
- Returns the tool-generated answer

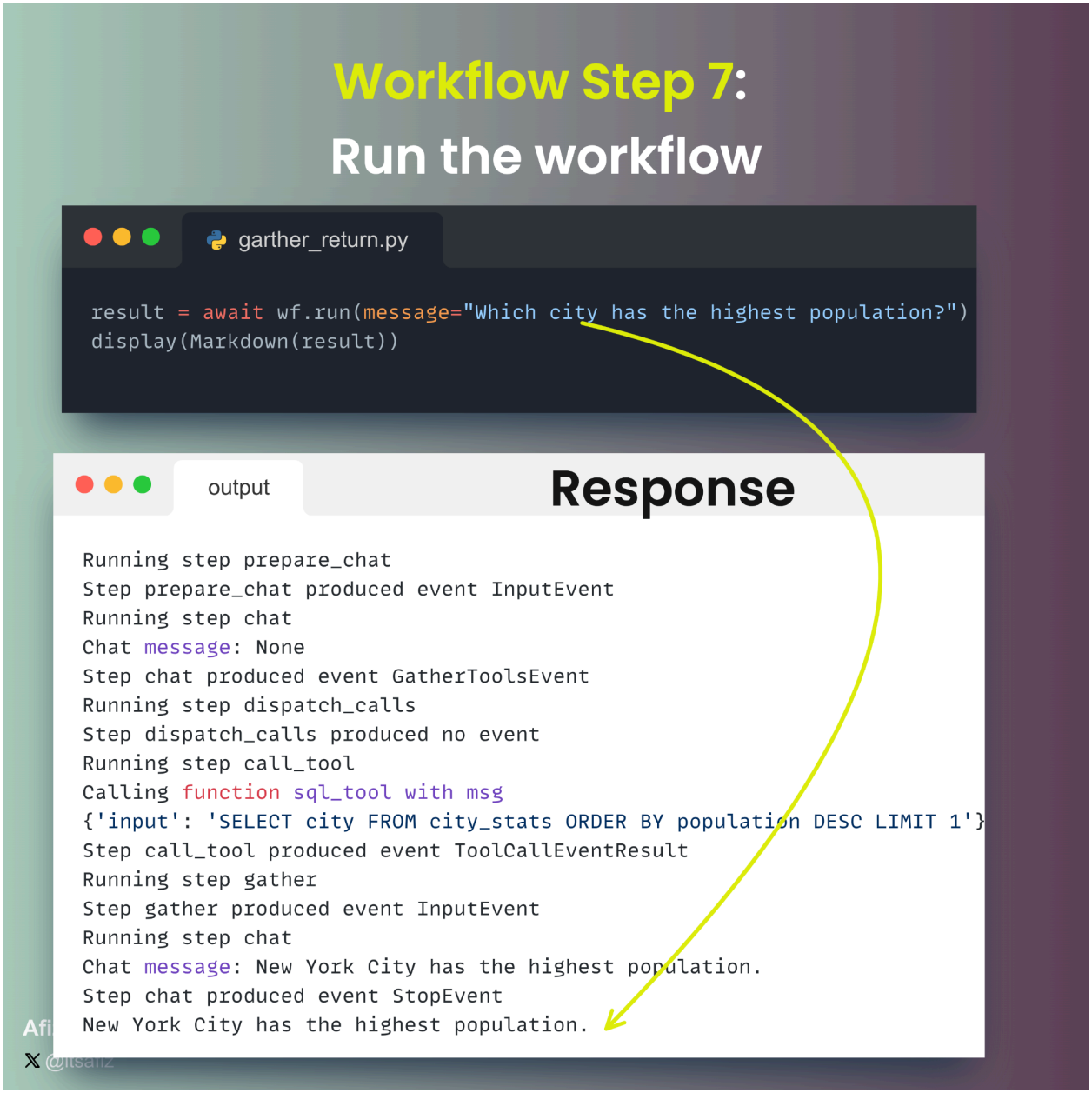
7. Initialize the workflow

Creates an instance of `RouterOutputAgentWorkflow` with:

- `sql_tool`: SQL query engine for database searches.
- `llama_cloud_tool`: Llama Cloud AI tool for answering questions.

8. Run the workflow

Run the workflow with a sample query and see the result 🙂



That’s all for today. Please checkout the GitHub for complete code along with Streamlit UI.

https://github.com/afizs/ai-engineering-hub/tree/main/llamacloud_sql_router