**Question – Answering Agent For SQL Database**

**How to Install Ollama :**

## For Linux Install with one command :

## curl -fsSL https://ollama.com/install.sh | sh

For Windows use this Link : <https://ollama.com/download/windows>

How to Download LLM Model :

Download any One Source LLM Model From :

* + - * + [HUGGINGFACE](https://huggingface.co/)

Download Mistral-7B :

https://huggingface.co/TheBloke/Mistral-7B-Instruct-v0.2-GGUF/resolve/main/mistral-7b-instruct-v0.2.Q4\_0.gguf

Setting Up LLM Locally:

 **Download Model File**

* Ensure you have downloaded mistrallite.Q4\_K\_M.gguf to a directory, e.g., ./downloads/.

 **Install Ollama**

* Install Ollama :

## curl -fsSL https://ollama.com/install.sh | sh

 **Create Ollama Model**

* Use the Ollama CLI to create the model instance:
* ollama create anyname -f model filepath (.gguf file)

ollama create mistrallite -f ./downloads/mistrallite.Q4\_K\_M.gguf

* + mistrallite: Name of the model instance to create.
  + -f ./downloads/mistrallite.Q4\_K\_M.gguf: Path to the downloaded model file.

 **Verify Installation**

* Check if the model appears in the list of available models:

**Type Terminal:**

ollama list

 **Usage**

* You can now use the mistrallite model instance within Ollama for your natural language processing tasks.

Import Statements:

 from langchain.utilities import SQLDatabase

**Purpose:** Handles SQL database interactions.

 from langchain\_experimental.sql import SQLDatabaseChain, SQLDatabaseSequentialChain

**Purpose:** Create chains for interacting with SQL databases.

 from langchain.prompts import PromptTemplate

**Purpose:** Create templates for prompts.

 from langchain.prompts.chat import HumanMessagePromptTemplate

**Purpose:** Create templates for human messages in chat interactions.

 from langchain\_core.prompts import PromptTemplate

**Purpose:** Create and manage prompt templates.

**PostgreSQL Database Connection using psycopg2 and SQLAlchemy :**

Import Statements:

* import psycopg2

**Purpose:** Used to connect to PostgreSQL databases using the psycopg2 library

 from sqlalchemy import create\_engine, text

**Purpose:** create\_engine is used to create an SQLAlchemy engine, and text is used to write SQL queries.

 from sqlalchemy.orm import sessionmaker

**Purpose:** sessionmaker is used to create a configured "Session" class.

 from urllib.parse import quote\_plus

**Purpose:** Used to URL encode the password to safely include special characters.

**CODE FLOW:**

1. **Import Necessary Libraries**:
   * Imports psycopg2 for direct PostgreSQL database connection.
   * Imports from sqlalchemy for ORM-based operations.
   * Imports quote\_plus from urllib.parse for encoding the password.
2. **Connection Details**:
   * Defines username, password, host, port, and mydatabase for PostgreSQL connection.
3. **URL Encode Password**:
   * Uses quote\_plus to encode the password for inclusion in the SQLAlchemy connection URI.
4. **Connecting with psycopg2**:
   * Tries to connect to PostgreSQL using psycopg2.
   * Prints a success message upon successful connection.
   * Handles exceptions if connection fails.
5. **SQLAlchemy Connection URI**:
   * Constructs a connection URI for SQLAlchemy using the encoded password and connection details.
   * Prints the SQLAlchemy connection URI for debugging purposes.
6. **Creating SQLAlchemy Engine and Session**:
   * Tries to create an SQLAlchemy engine using the constructed URI.
   * Defines a session using sessionmaker bound to the engine.
   * Executes a sample query (SELECT version();) to verify the SQLAlchemy connection.
   * Prints a success message upon successful connection.
   * Handles exceptions if SQLAlchemy setup or query execution fails.
7. **Closing psycopg2 Connection**:
   * Closes the psycopg2 connection to release resources after use.

**Creating LLM Using Ollama :**

1.**Prompt Template Definition**

* Defines a template for generating prompts that constructs a syntactically correct PostgreSQL query based on an input question.

2.**Imports**

* Imports necessary libraries and modules including:
  + Ollama from langchain\_community.llms for language model interaction.
  + Various utilities from transformers, datasets, peft, langchain, and others for NLP tasks and database interactions.

3.**Initialize Ollama Language Model**

* Initializes an instance of the Ollama language model (ollama) using the "mistrallite" model configuration.

4.**Define and Initialize Prompt Template**

* Creates a prompt template (tweet\_prompt) from the defined template for generating SQL queries.

5.**Create LLMChain for Tweet Generation**

* Sets up an LLMChain (tweet\_chain) for generating SQL queries based on the Ollama model and the defined prompt template, with verbose logging enabled.

6.**Initialize SQLDatabaseChain**

* Creates an instance of SQLDatabaseChain (db\_chain) for interacting with an SQL database using the Ollama model, a specified database (db), the prompt template, and verbose logging, with a top-k limit for query results.

7.**Initialize SQLDatabaseSequentialChain**

* Sets up a SQLDatabaseSequentialChain instance (db\_chain1) for sequential database interactions using the Ollama model, a specified database (db), and verbose logging.

8.**Invoke a Query**

* Executes a specific query (Give all values present in company\_id column of account\_account table) against the SQL database using db\_chain1.