Assessment - 2

AI Agent Coding Assessment: Intelligent Query Router

Objective

The goal of this assessment is to design and implement an **Intelligent Query Router** that efficiently determines the best source (SQL database or Vector DB) to fetch relevant answers based on confidence scoring. The agent should not rely solely on predefined rules but should make dynamic decisions using a **confidence-based ranking system**. Additionally, if neither data source provides high-confidence results, the agent should **rephrase the query** and retry intelligently.

Scenario

You are developing a **knowledge assistant** capable of processing natural language queries and retrieving relevant data from multiple sources. The agent should decide:

1. Whether **SQL database results are sufficient** to answer the query confidently.
2. If SQL is insufficient, whether **Vector DB embeddings search** provides a more relevant response.
3. If **both sources are weak**, the agent should **rephrase the query** and attempt again.

The system should be efficient, making minimal queries while maximizing relevance.

Key Functionalities

1. Query Processing

- Preprocess the user's input.
- Perform basic **NLP-based keyword extraction**.
- Normalize the query to remove unnecessary stopwords and variations.

2. Decision-Making with Confidence Scoring

- Query the **SQL database** first and retrieve results.
- Compute a **confidence score** based on SQL results using factors such as:
  - Completeness of the response (e.g., are key fields missing?)
  - Data relevance (e.g., does it match the intent?)

- Then perform a **Vector DB search**:
  - Extract **keywords or embeddings** from the query.
  - Compute similarity scores using a hybrid approach (e.g., **BM25 + embeddings similarity**).
  - Compare **SQL confidence** vs. **Vector DB similarity score**.

3. Query Rephrasing & Retry

- If neither data source provides a **high-confidence response**, the agent should **intelligently rephrase** the query and retry once.
- Rephrased queries should be **semantic equivalents**, not just simple synonym substitutions.
- Possible approaches:
  - Use an **LLM-based rephraser** to generate alternatives.
  - Apply **query expansion techniques** (e.g., add missing context, use similar phrasing).

Technical Requirements

1. Core Implementation

- Use **LangGraph** for modular and scalable agent design.
- Implement SQL queries using **PostgreSQL / MySQL / SQL LITE**
- Use a **Vector DB** like ChromaDB
- Design a confidence scoring mechanism for decision-making.

2. Optional Enhancements

- **Query Caching**: Optimize performance by caching frequently asked queries.
- **LLM Integration**: Use an LLM for intelligent query rephrasing.
- **Hybrid Ranking**: Combine **BM25 scoring** and **embedding similarity**.

Submission Guidelines

- Provide a **README.md** explaining your approach, decision-making logic, and setup instructions.
- Ensure your code is modular and well-documented.
- If implementing the **API**, include sample queries and expected outputs.

Submission

Sample Question 1 with its selected Database SQL or Vectorstore.

```
{
  "check_query_type": {
    "query": "when kkr bought Brendon  McCullum?",
    "query_type": "structured"
  }
}
---
Current step output:
{
  "preprocess": {
    "query": "when kkr bought brendon mccullum?",
```

Output :

```
---
Current step output:
{
  "rerank": {
    "best_source": "Vector"
  }
}
INFO:httpx:HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"
Current step output:
{
  "generate_response": {
    "response": "KKR bought back their former captain, Brendon McCullum, in the 2012 auction."
  }
}
---
```

Sample Question 2 with its selected Database SQL or Vectorstore.

```
Enter your query: How many runs do ganguly Have with his bat in cricket?
INFO:httpx:HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"
Current step output:
{
  "check_query_type": {
    "query": "How many runs do ganguly Have with his bat in cricket?",
    "query_type": "structured"
  }
}
---
Current step output:
{
  "preprocess": {
    "query": "how many runs do ganguly have with his bat in cricket?",
```

Output :

```
---
Current step output:
{
  "rerank": {
    "best_source": "SQL"
  }
}
INFO:httpx:HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"
Current step output:
{
  "generate_response": {
    "response": "The information provided in the SQL knowledge base shows that Ganguly has scored 1349 runs with his bat in cricket."
  }
}
---
```