# Langchain

## Quickstart: LangChain Essentials – Python

***Why use langchain?***

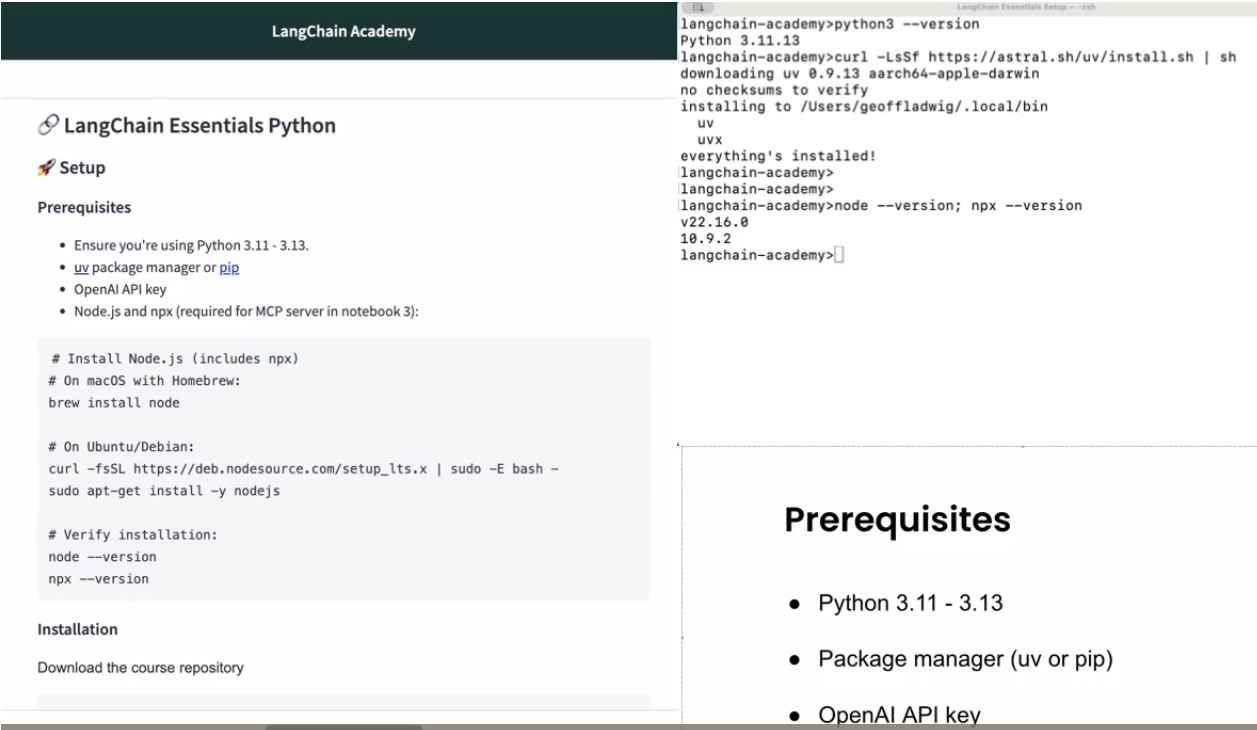
* Langchain is Open source library.
* Langchain used for build the LLM Applications
* Langchain is model independent- it will work for all the models -Just you have import the specific LLM model and change the api key, remaining code is as it is.
* ***What is langchain Independent?***:Each LLM vendors has a unique api, these apis are being frequently updated, with langchain you can build your app and switch models between vendors and models without needing rewrite the code.
* We have recently added a new easy to use agent, that is really centerpiece this course., This agent build on Langgraph giving the agent is durable execution, persistence and more. The agent itself has been built to be extremely flexible, in particular, it support what we call middleware .this enables you to customize your gent to fit your needs.
* Learn- Here:

Building blocks- Models, messages, memory and tools.

You will use the agent and enhance it with the middleware. This is will let you add dynamic prompting and model selection, adapting the prompt and model to changing the conditions, you will learn how to add a custom before and after model and tool calls.

Here you will use the langsmith and studio a visibility tool specifically designed for stateful agents.

## Getting started:



## Langchain-Document:



## Course Transcripts



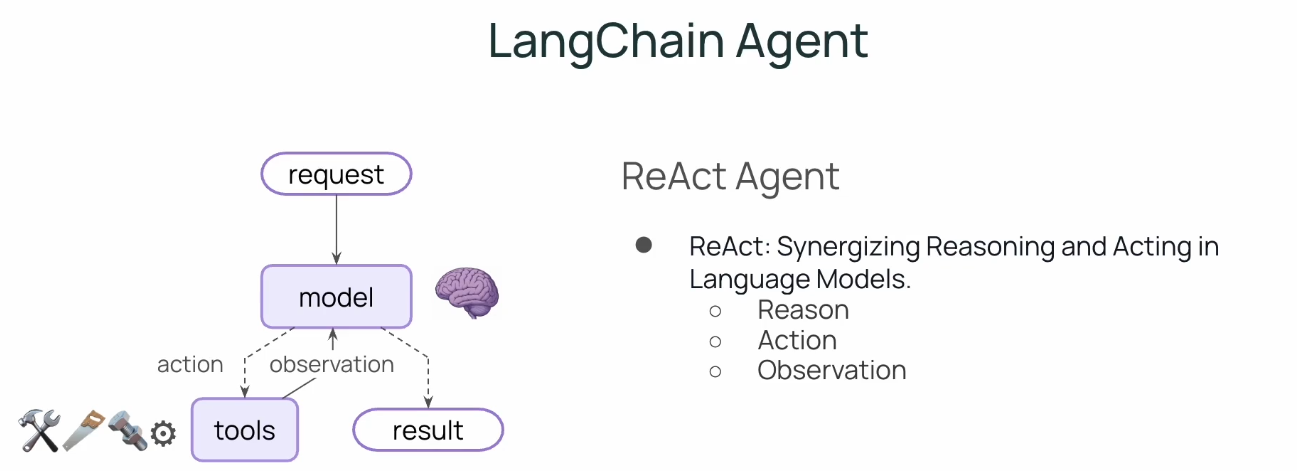
# Lessons:

## Create Agent:

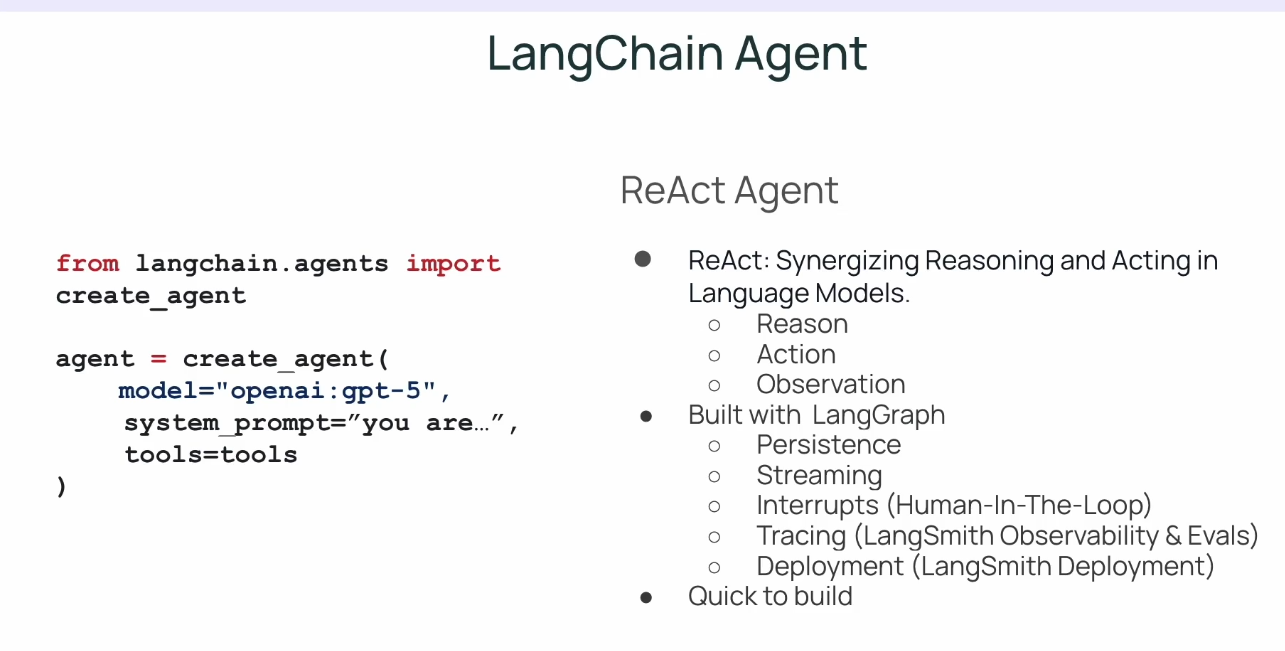
Middles ware : to add pre-built or customized features.

ReAct-Agent-

* ReAct meansRe-Reasoning Act- Acting .
* ReAct Agents operate in a loop, it starts when a model receives a request, it first reasons about it, then it uses tools to take action and the tool provide observations, At this point, the LLM can reason, determine if it can respond to the request by producing a final result, or it can take further action and continue the tool calling loop.



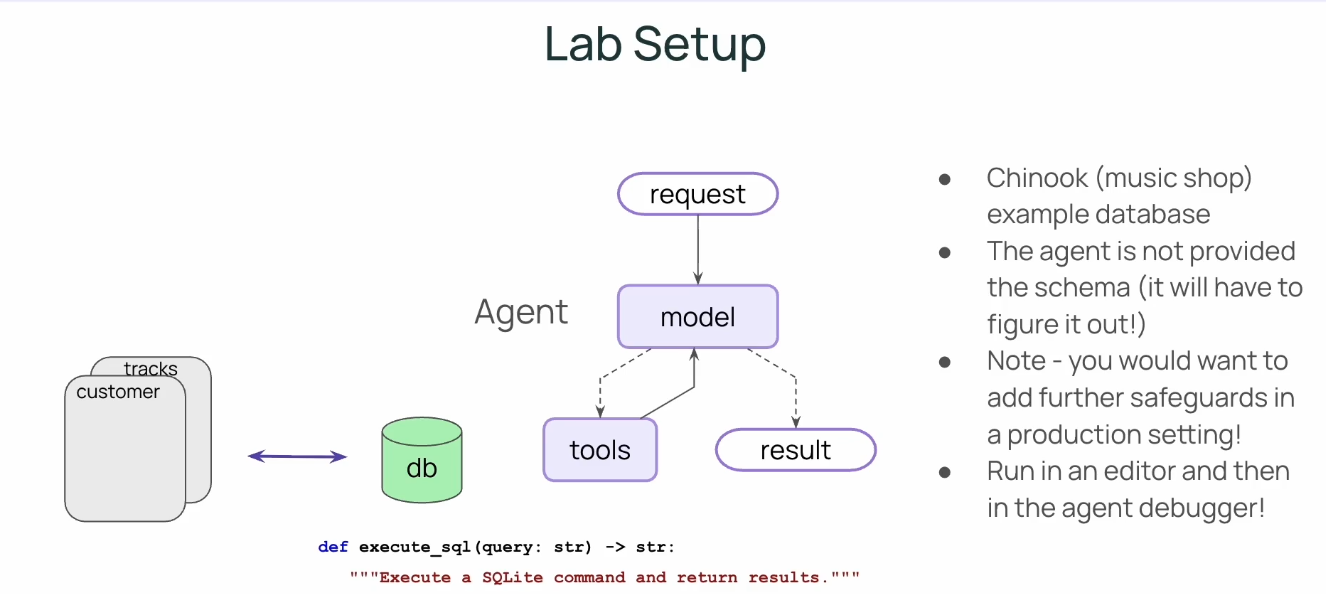
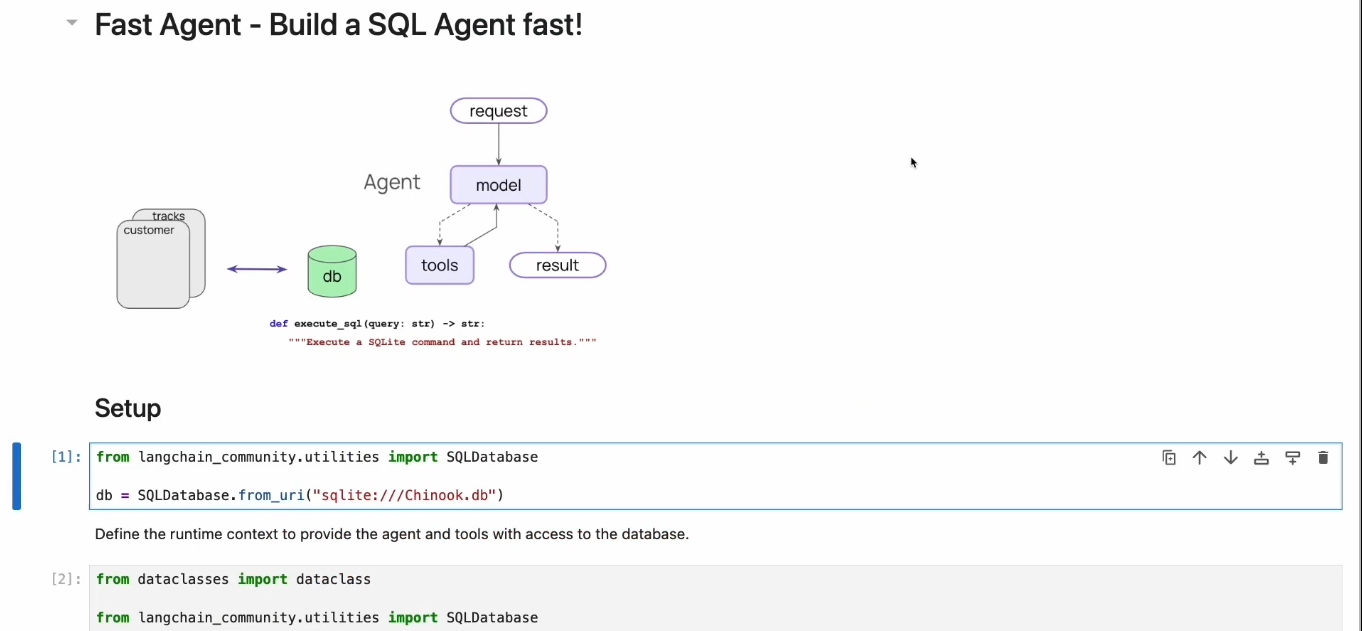
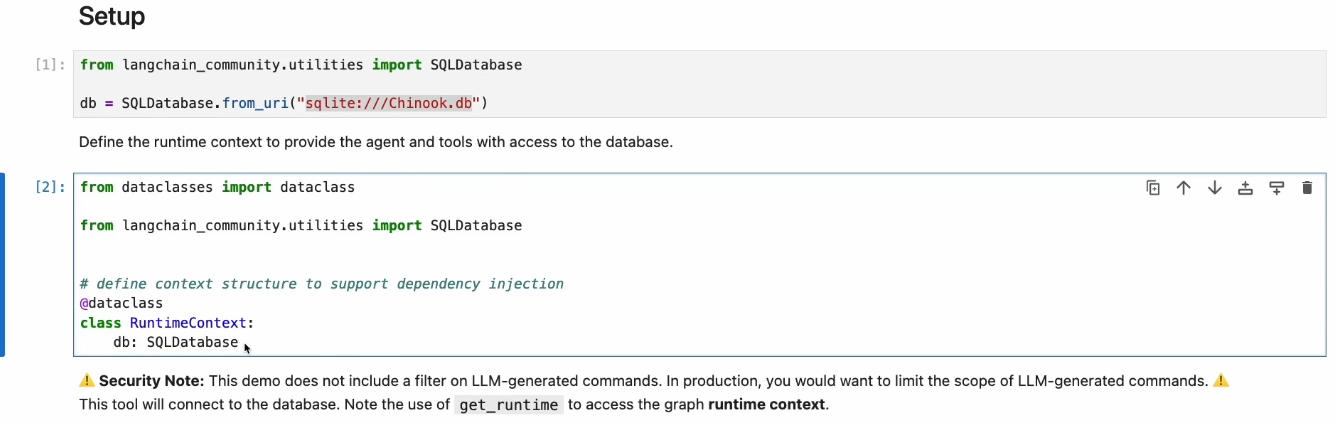
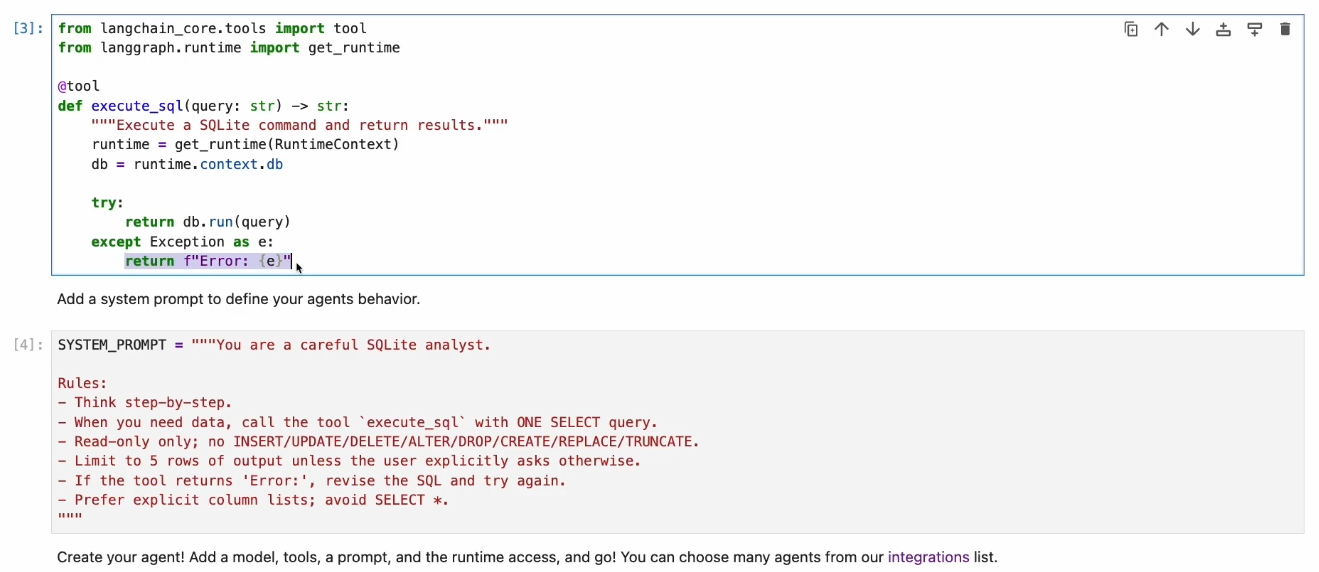
**Synergizing:** means working together of two things( ex: muscels or drugs) to produce an effect greater than the sum of their individual effects.



The new model is built with langgraph under the hood, which provides persistence and durability, streaming and interrupts, tracing and deployment.

## Lab-1

Ex: Single tool used to execute the sql lite commands .

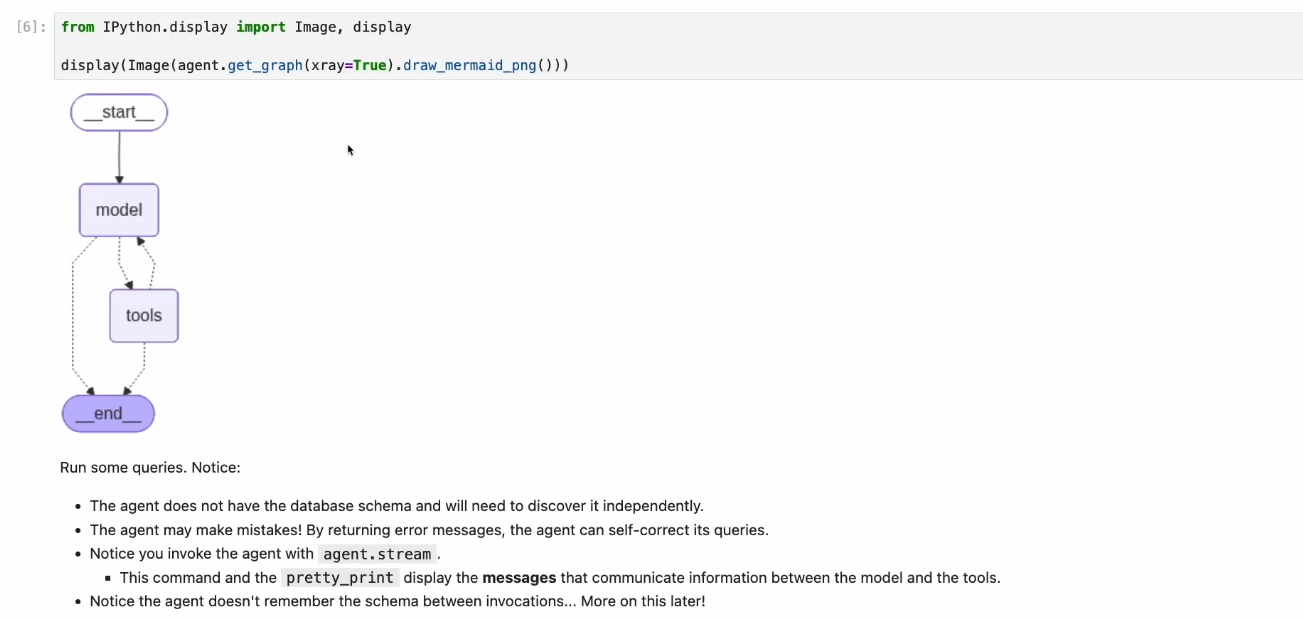


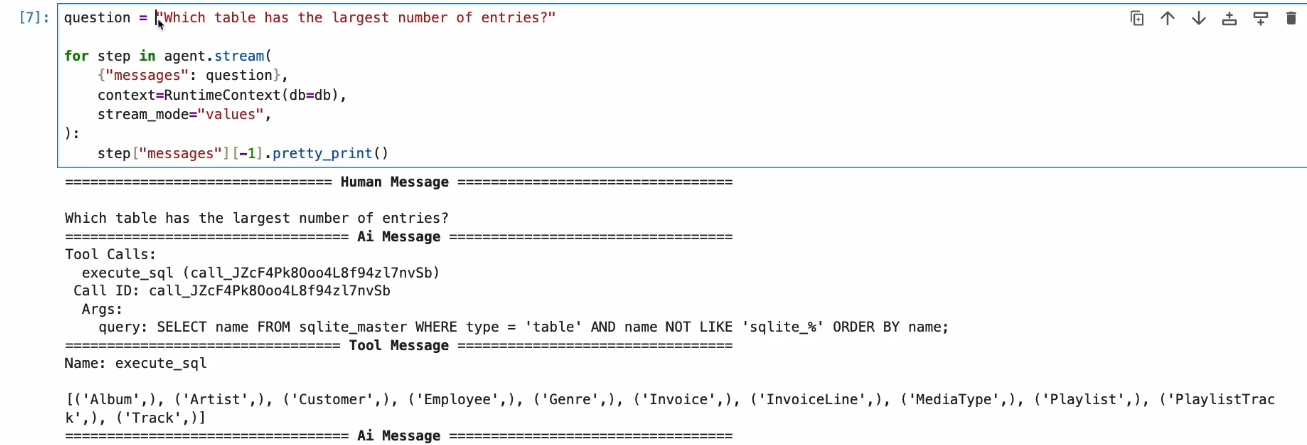
Here RuntimeContext: used to use access to DB

Tools: Add custom Function as Execute\_SQL

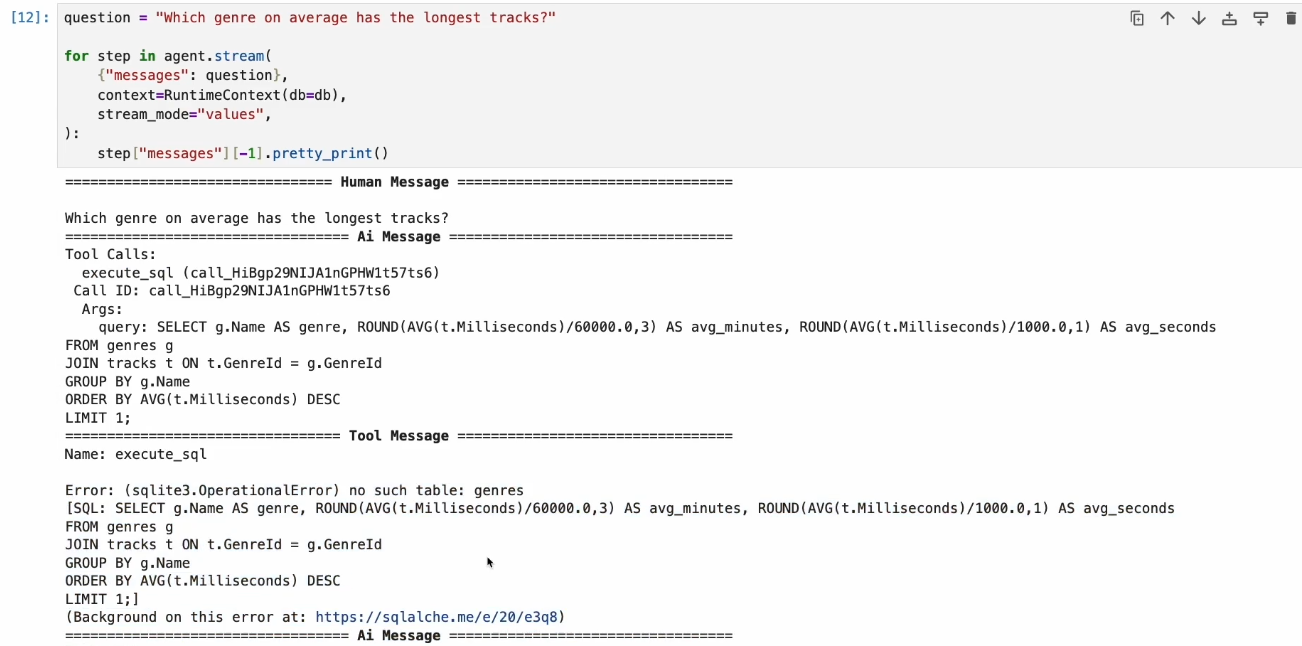
Model – LLMs

System\_Prompt:

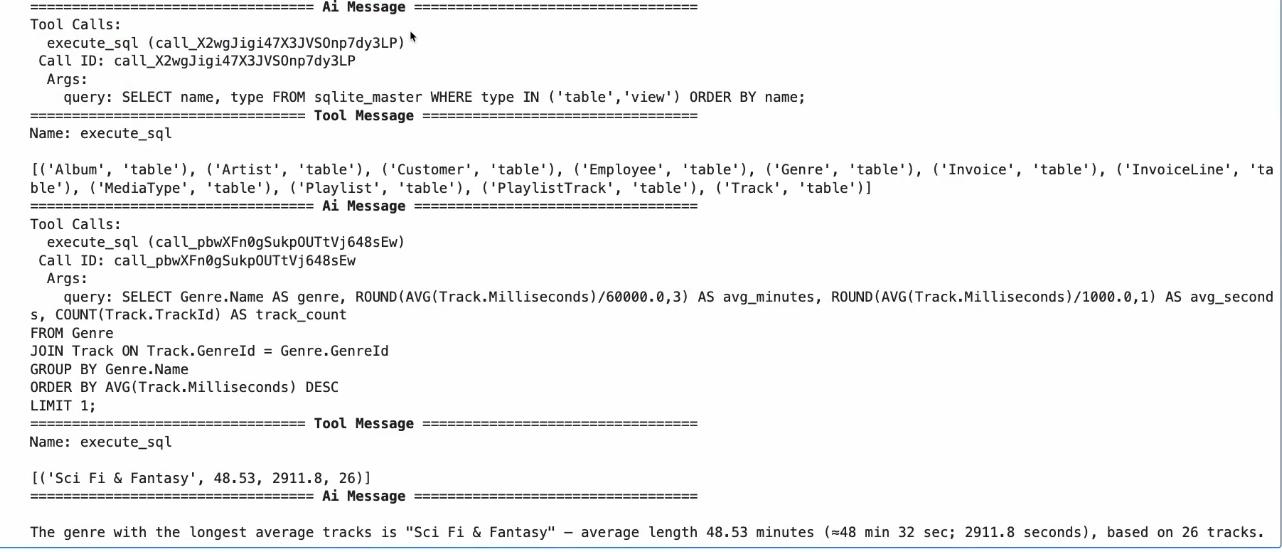


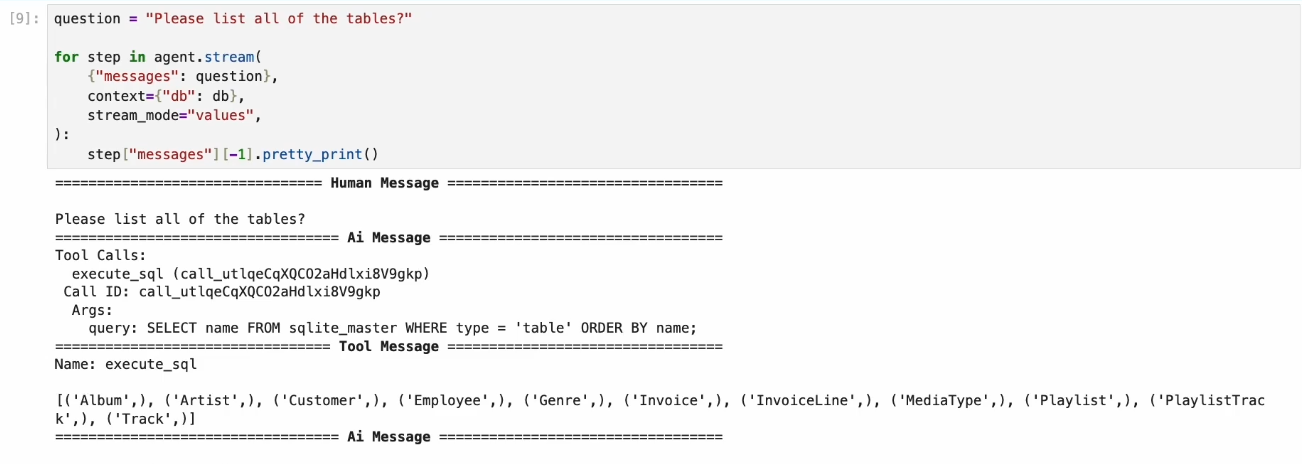
<https://academy.langchain.com/courses/take/langchain-essentials-python/lessons/69388275-lesson-1-create-agent>

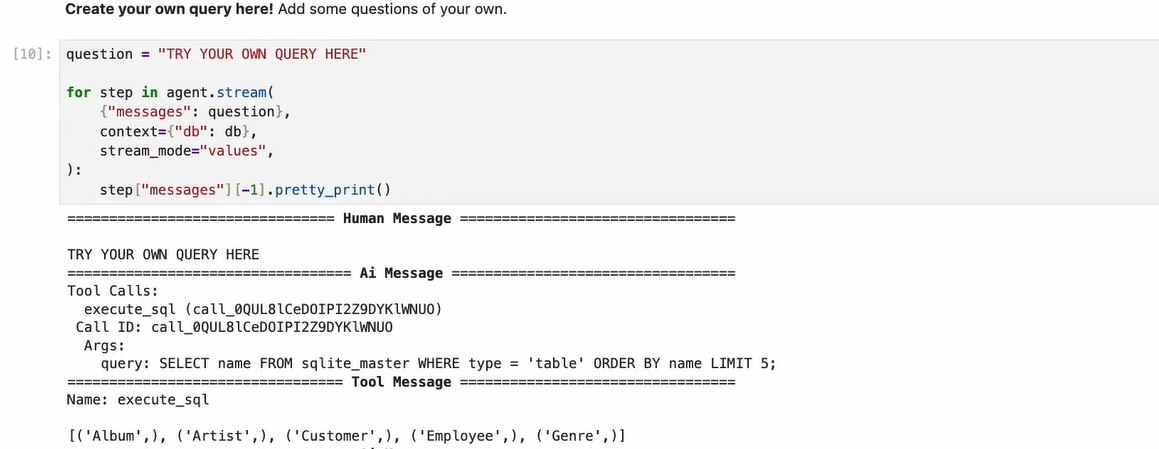


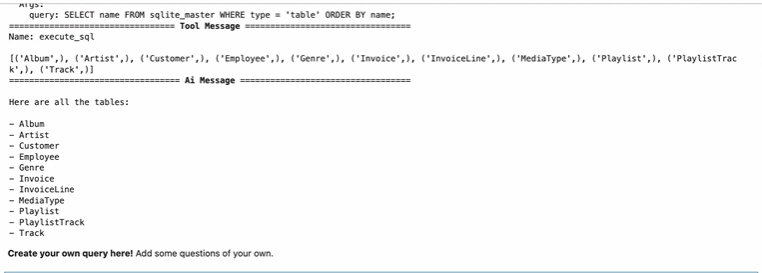
Agent doesn’t have access to DB/Schema, It does not remember which tables we have from the previous conversation.



Above screen shot ai again looped, requesting the tables and check the genre table in the result and then execute the sql result and then we see a successful query on genre table.

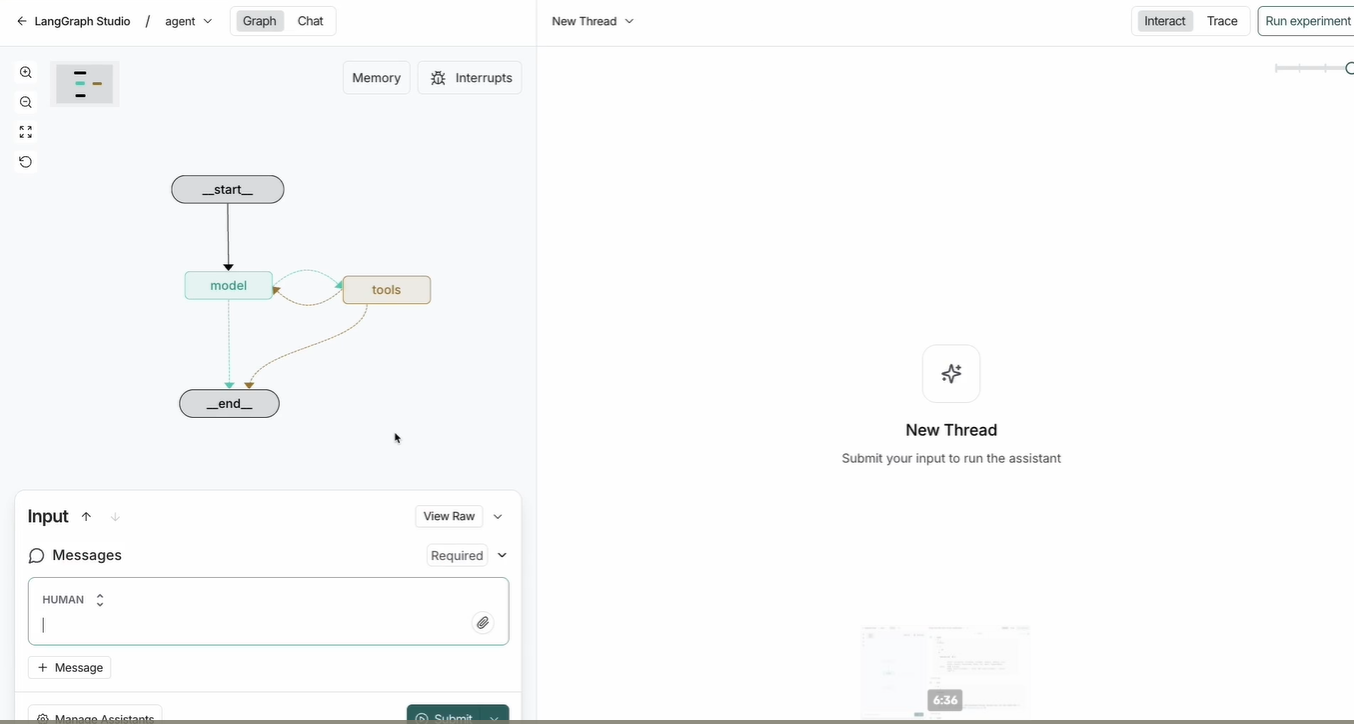
 



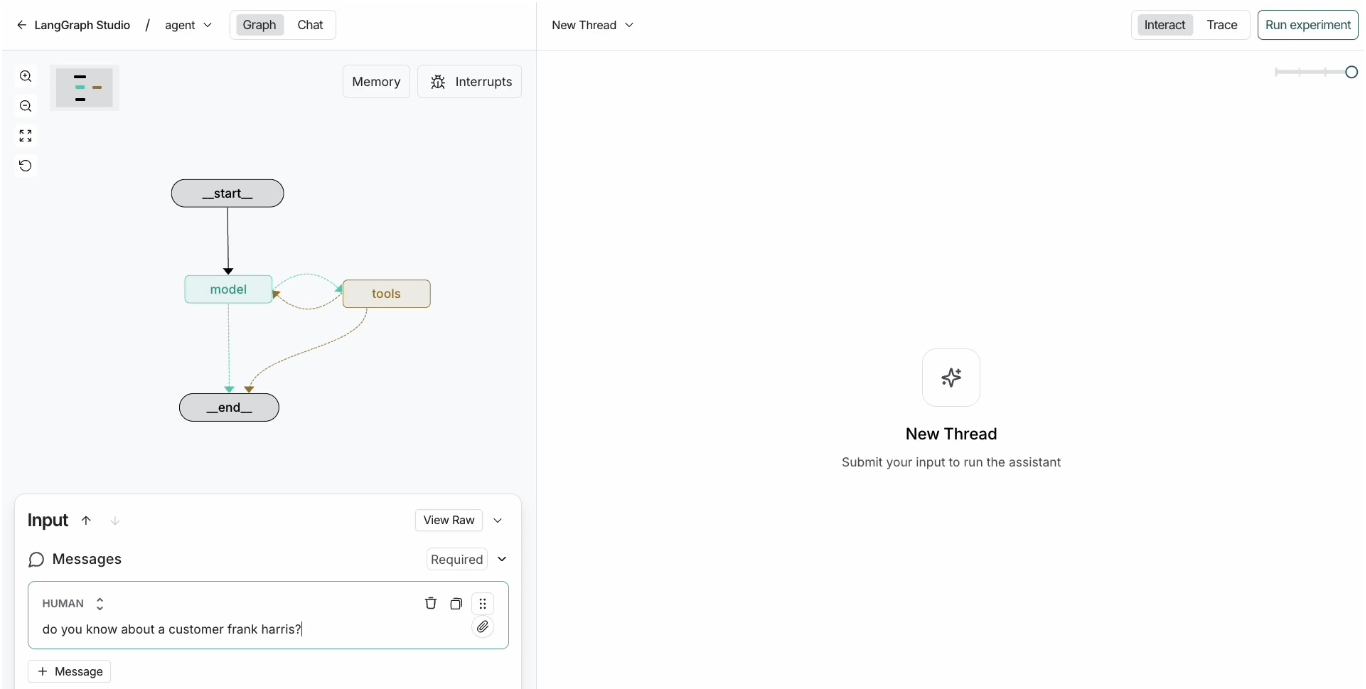
Lets try in Debugger:

Welcome to an agent debugger;

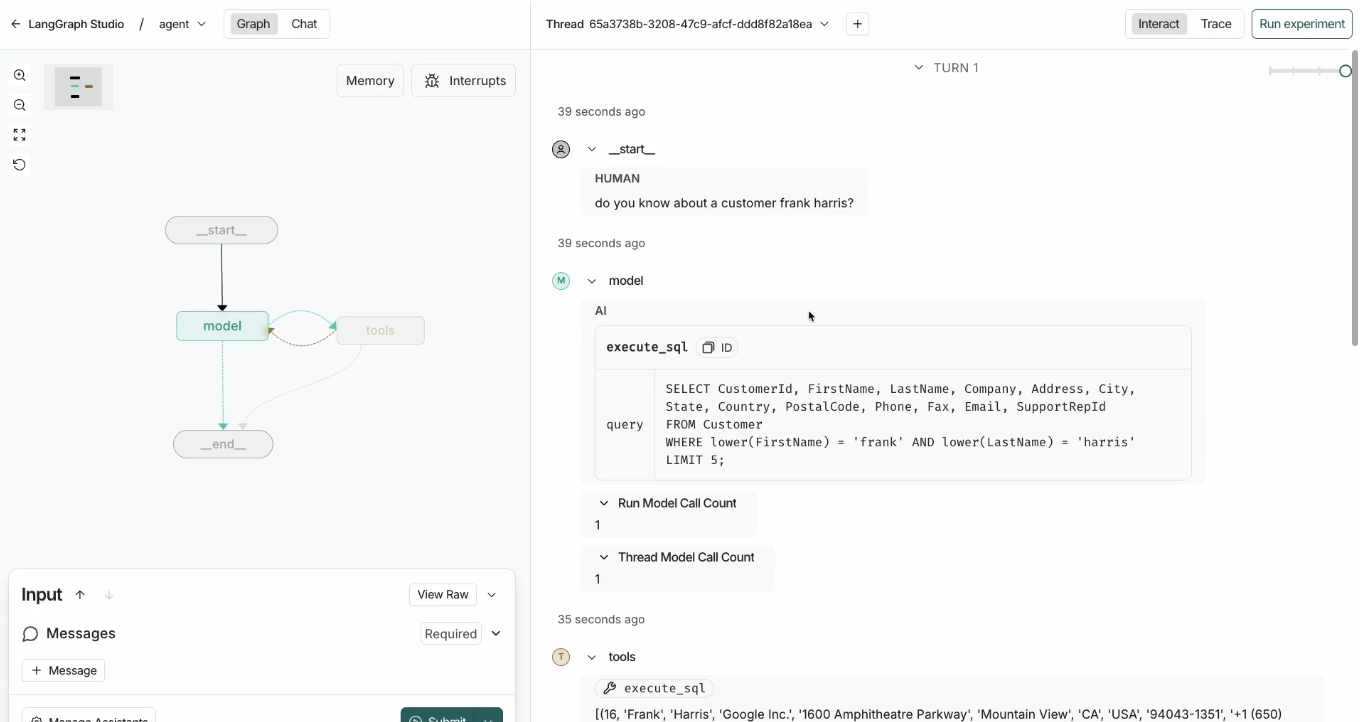
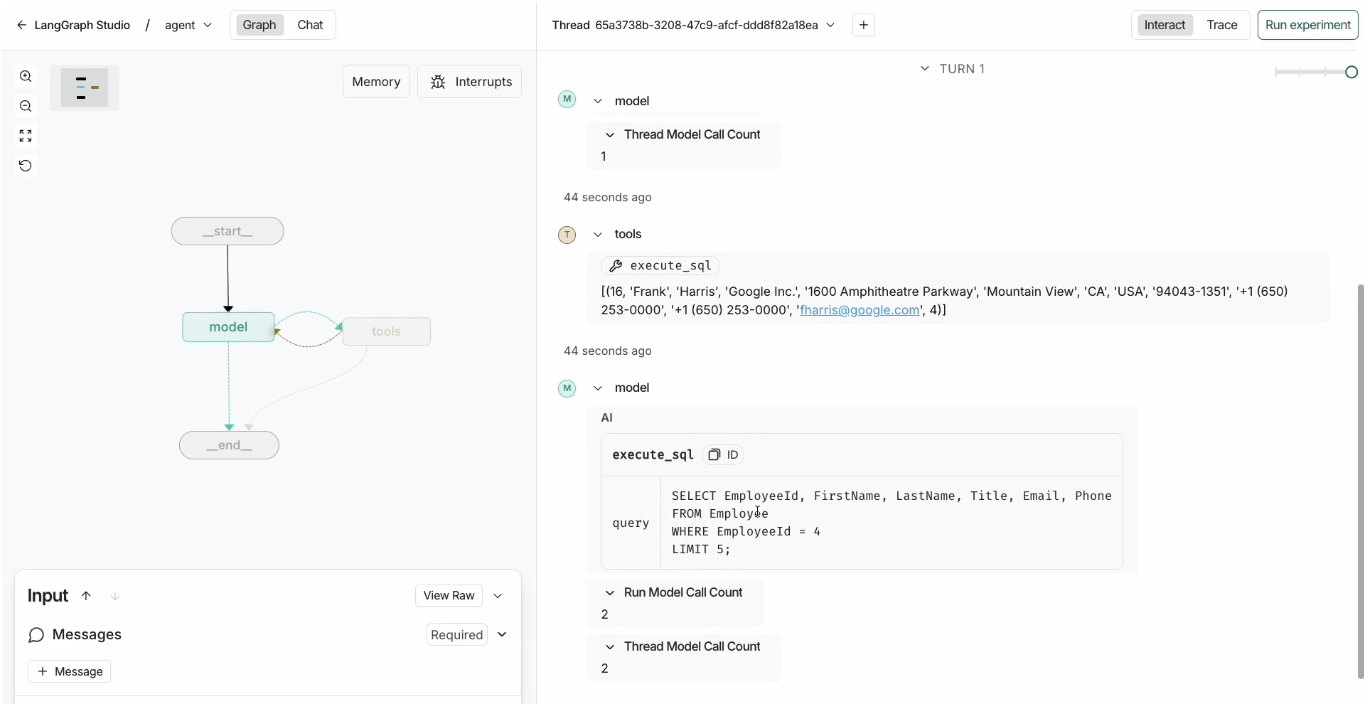
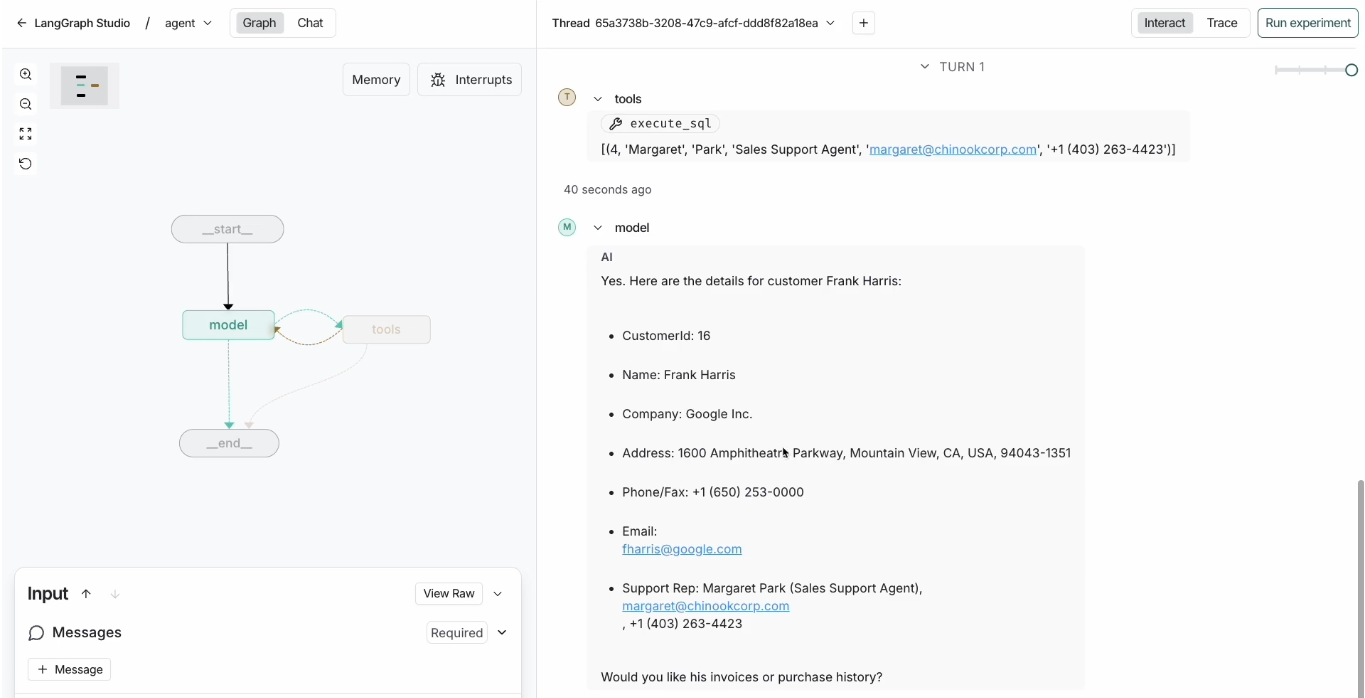


In Agent Debugger – Classical Graph representation of Agent Here with a model and tools calling loop.

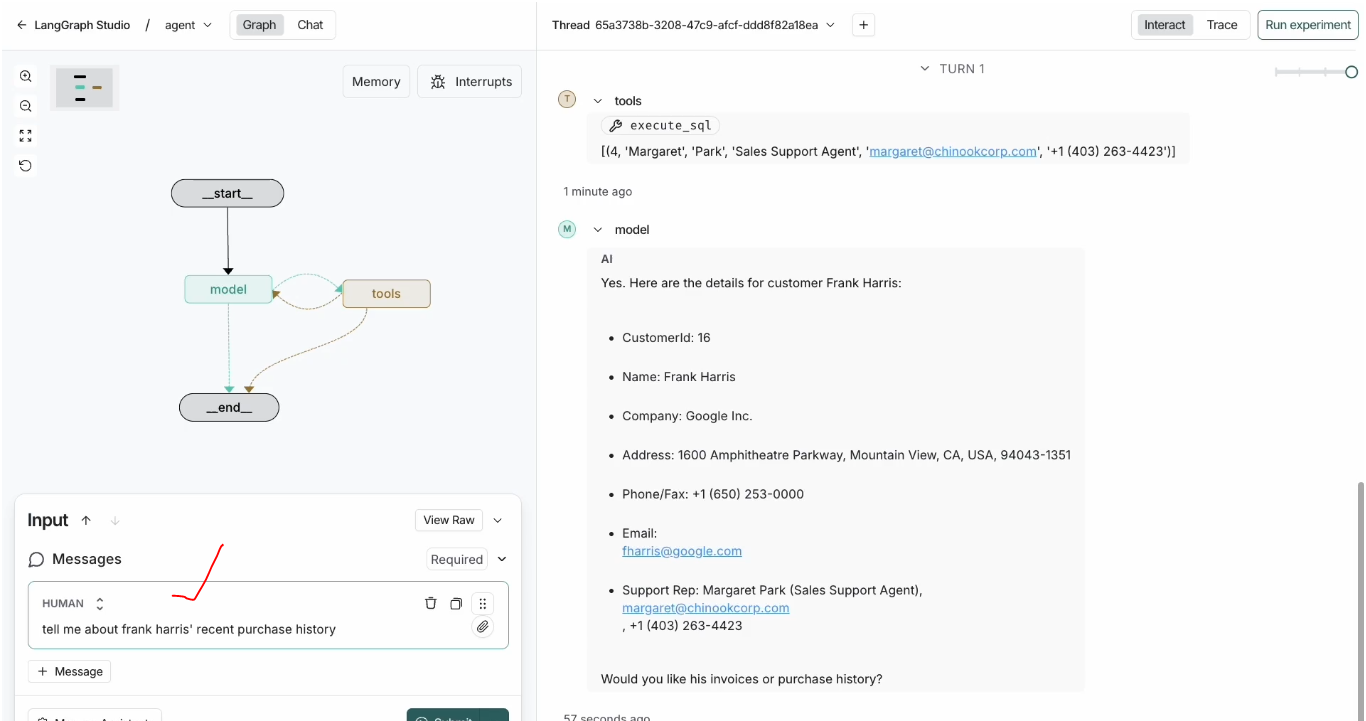
Lets try asking a question:



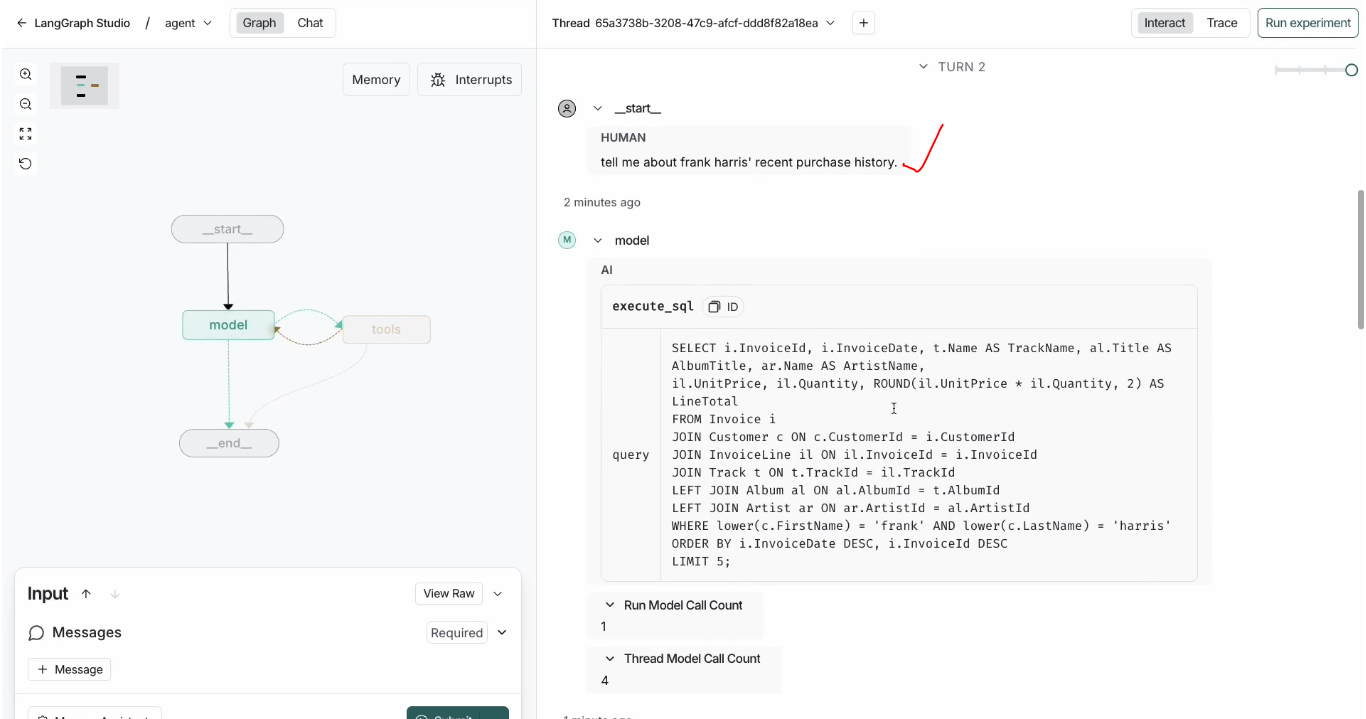
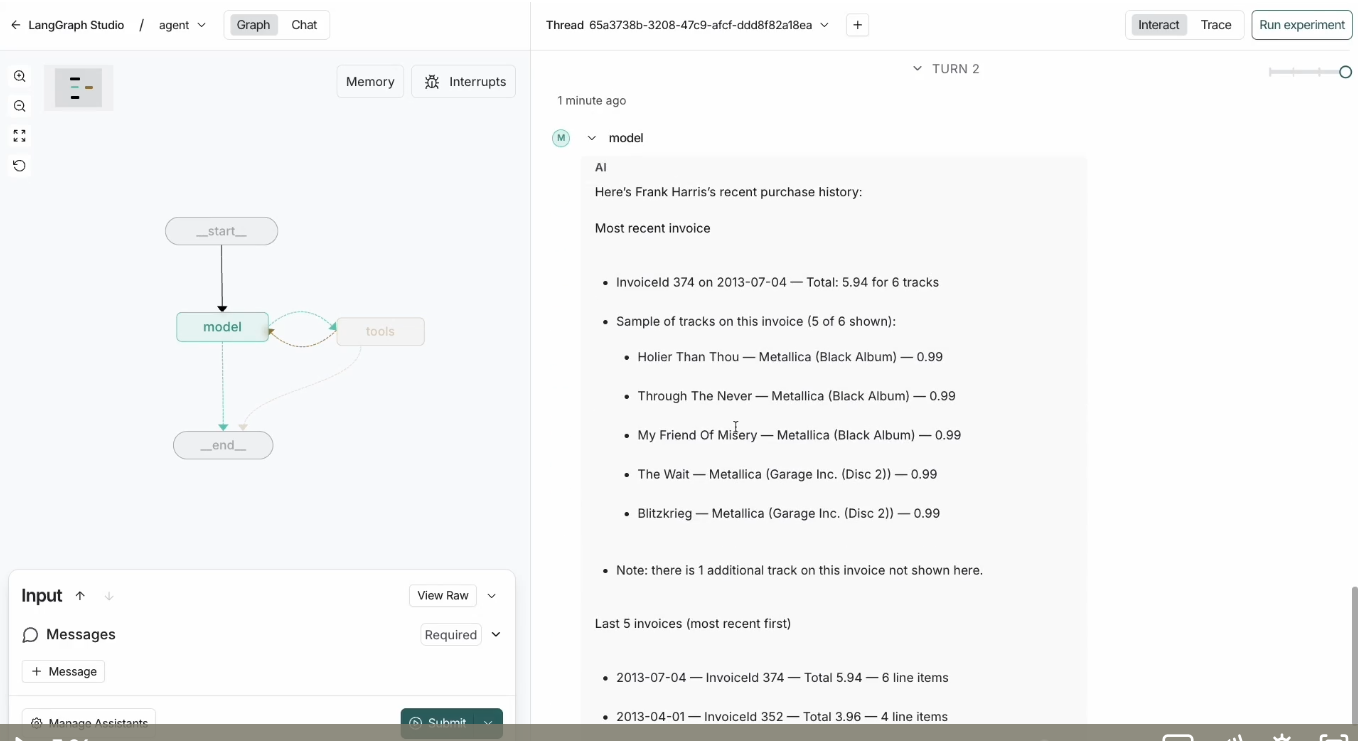
We can see Agent start and model has been executing.

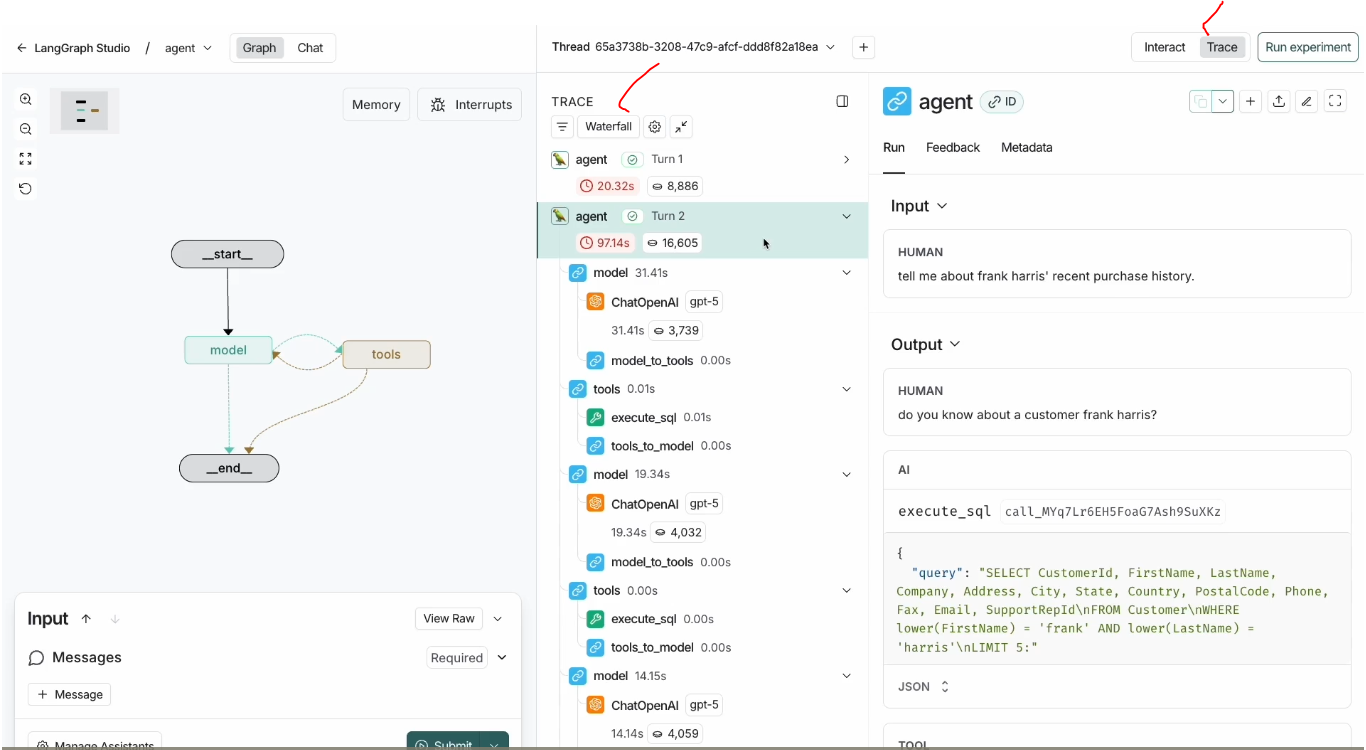
Lets ask another question:



Execution of Agent in Debug:

Trace mode:



Code DB Run with Agent:

