Function Calling with LLMs
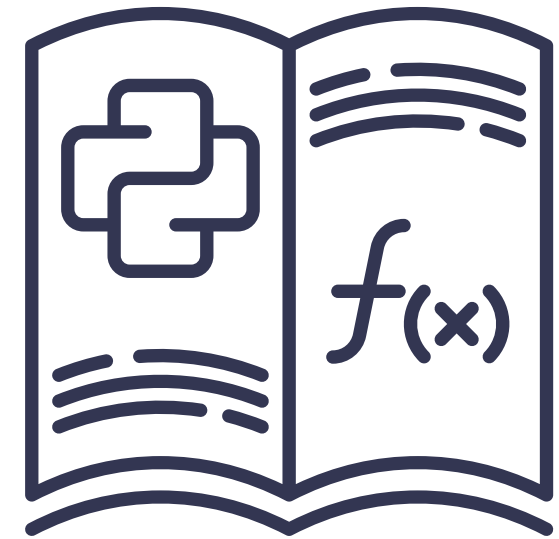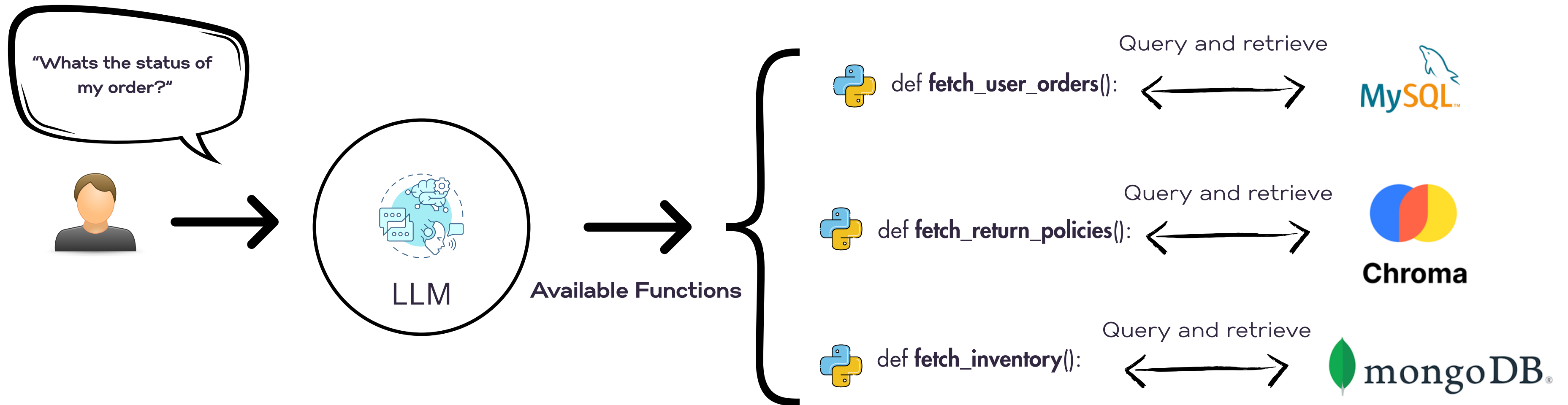
- Function calling allows large language models (LLMs) to interact with external tools and APIs by intelligently determining when and how to call specific functions with appropriate parameters.
  - External APIs
  - Databases
  - Search engines
  - **Python Functions**

- Function calling involves providing the LLM with a description of available functions, including their
  - *Function Names*
  - *Input Arguments with data types*
  - *Expected outputs with data types*

- In **Function calling** the **LLM** analyzes user prompts and determines if a function needs to be called to fulfill the request.
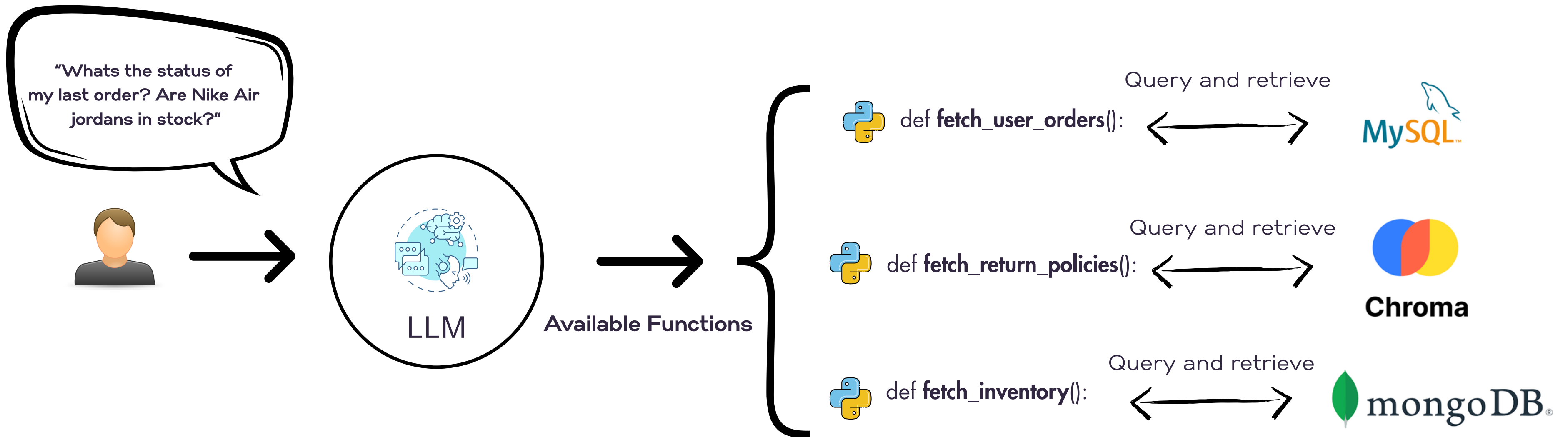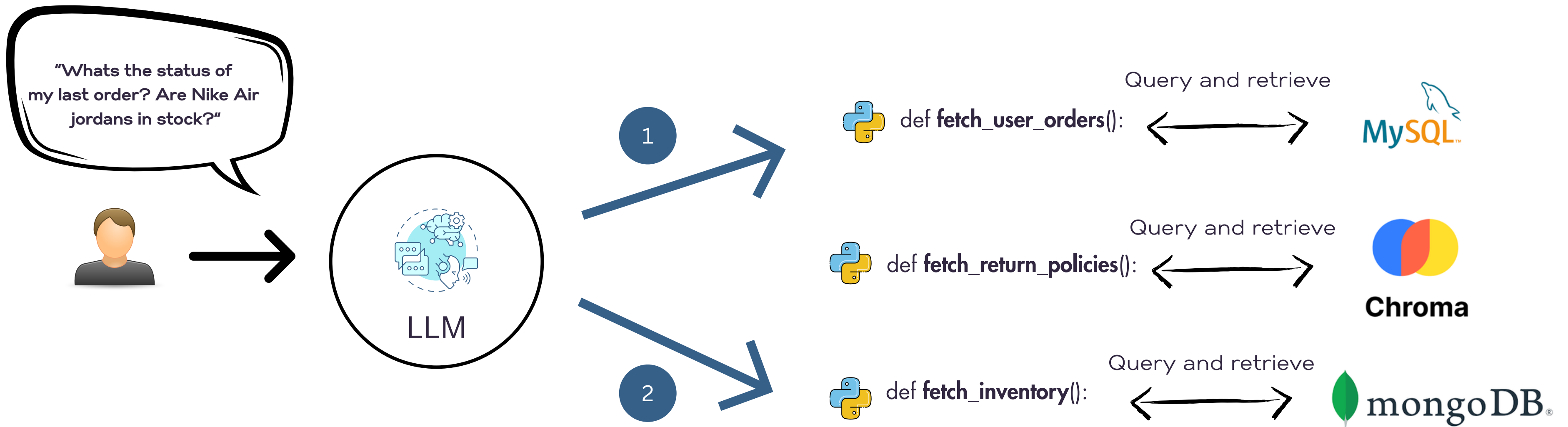
Function Calling with LLMs

Function Calling with LLMs

Function Calling with LLMs

"Whats the status of my last order? Are Nike Air jordans in stock?"

LLM

Available Functions

def **fetch_user_orders**():

Query and retrieve

MySQL

def **fetch_return_policies**():

Query and retrieve

Chroma

def **fetch_inventory**():

Query and retrieve

mongoDB.

Function Calling with LLMs

"Whats the status of my last order? Are Nike Air jordans in stock?"

LLM

1

def **fetch_user_orders**():

Query and retrieve

MySQL

def **fetch_return_policies**():

Query and retrieve

Chroma

2

def **fetch_inventory**():

Query and retrieve

mongoDB

Function Calling with LLMs
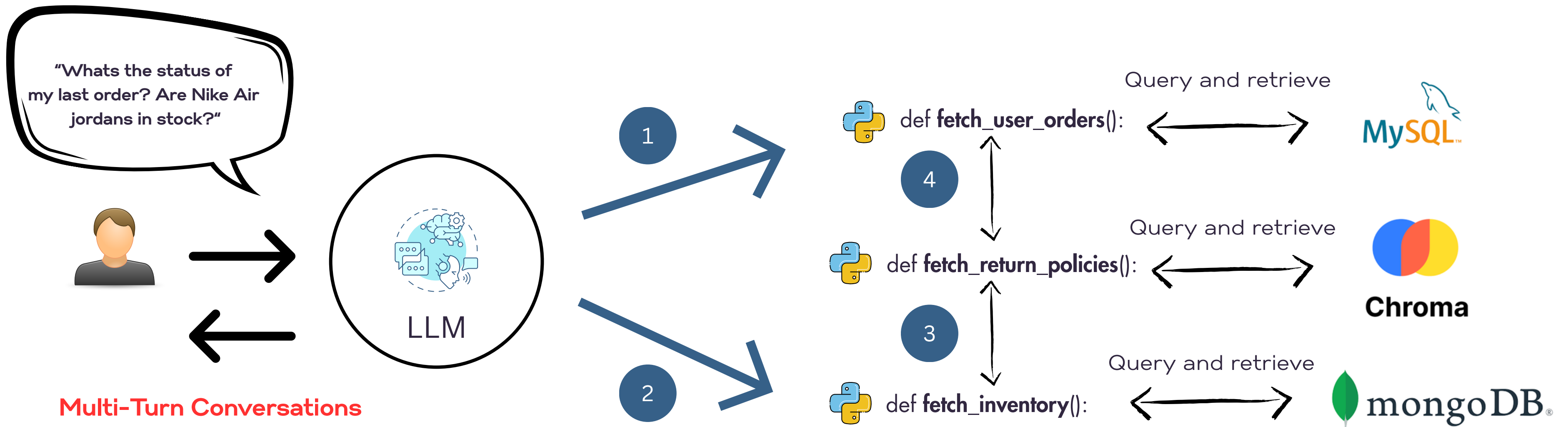
Function Calling with LLMs

▶ **Orchestration** is the process of coordinating various components to execute complex tasks.

Why is **orchestration** necessary with function calling?

- Not all data can/should be vectorized
- Extending Capabilities
- Enhancing Accuracy and Reliability
- Enabling Complex Workflows