SQL with Gen AI

Tables, Transformed with AI

Agenda

Application Overview	3
○ Current Flowchart	4
Data Overview	5
○ Tabular Database	5
○ Document Database	7
Agents Overview	11
○ SQL Agent	11
○ Analyst Agent	13
○ PDF Agent	15
Refactor Considerations	17

Application Overview

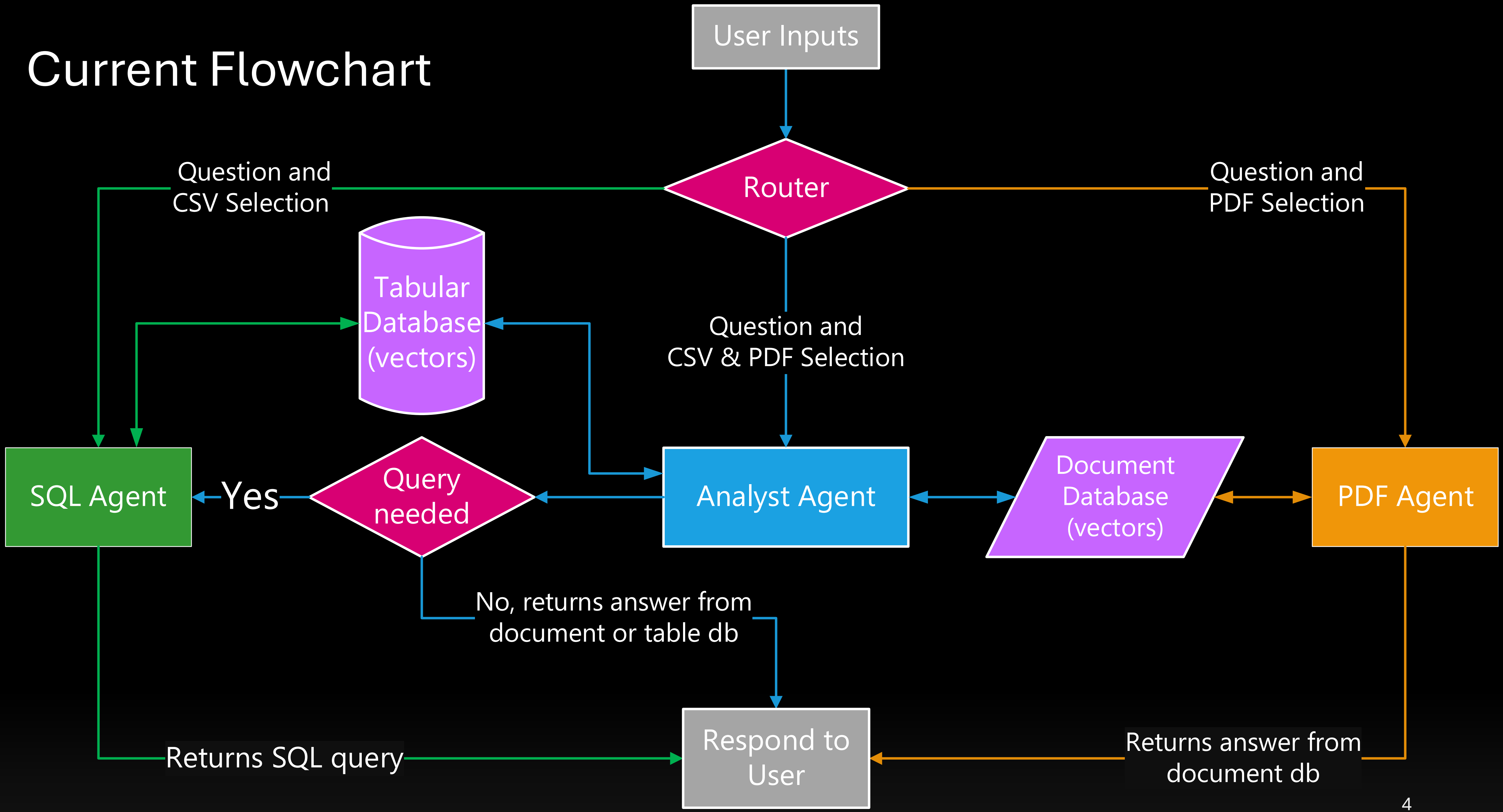
Objectives

Automate SQL query generation for efficient data retrieval, manipulation, and visualization

Core Concepts

- Leverage LLMs to translate natural language into database queries
- Retrieve relevant documents to provide context for query generation
- Implement an agentic communication pipeline for interactions

Current Flowchart



User Inputs
Router
Question and CSV Selection
Question and PDF Selection
Tabular Database (vectors)
Question and CSV & PDF Selection
SQL Agent
Yes
Query needed
Analyst Agent
Document Database (vectors)
PDF Agent
No, returns answer from document or table db
Respond to User
Returns SQL query
Returns answer from document db

Tabular Database

Tabular Database (vectors)

Source Data: CSV files

Data Manipulation

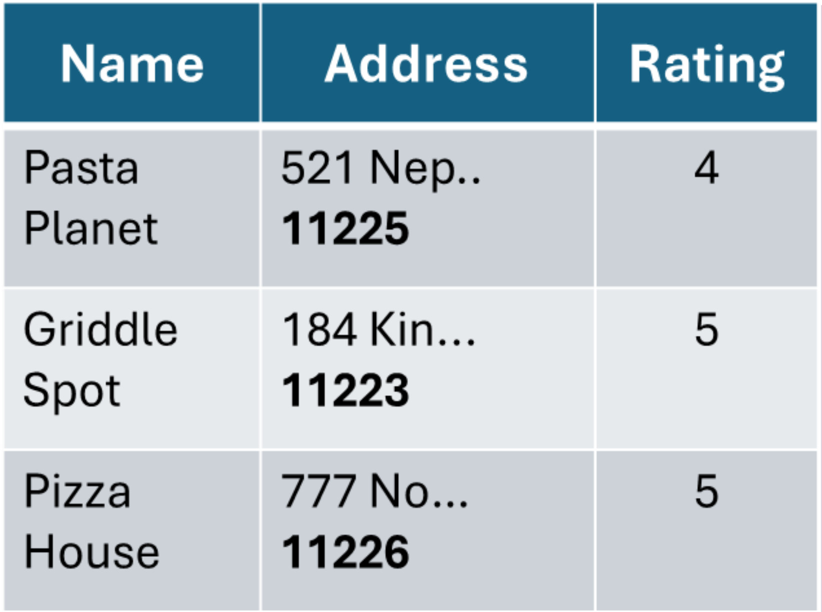
The CSV is loaded into a tabular database, with each row stored as a record. Row-level embeddings are generated and saved within the same database for similarity search.

Use cases

- The generated queries are intended for use on the tabular database
- Embeddings of rows provides valid data points for use in writing of queries

Chunking Strategy

Rows are converted into sentences using column names to provide semantic clarity



Access database			Create sentence per row	Create and store embedding per row
Name	**Address**	**Rating**		
Pasta Planet	521 Nep.. **11225**	4	1. The Name is Pasta Planet. The Address is 521 Napolean Rd, 11225. The Rating is 4.	{row_1: [embeddings...]}
Griddle Spot	184 Kin... **11223**	5	2. The Name is Griddle Spot. The Address is 184 Kindle Rd, 11223. The Rating is 5.	{row_2:[embeddings...]}
Pizza House	777 No... **11226**	5	3. The Name is Pizza House. The Address is 777 Nostrant, 11226. The Rating is 5	{row_3:[embeddings...]}

Document Database

Document
Database
(vectors)

Source materials: Organizations documentation, User manuals and textbooks

Data Manipulation

PDF is parsed into markdown text and then inserted into a Document Database

Use case

The document database contains supporting information that can provide added context that aids in query generation

Chunking Strategy

Problem with Traditional Vector Search

- Returns only the closest chunk
- No awareness of boundaries like Page, Section, or Chapter

Parent-Child Retriever Solution

Leverages graph relationships to define context-aware retrieval windows

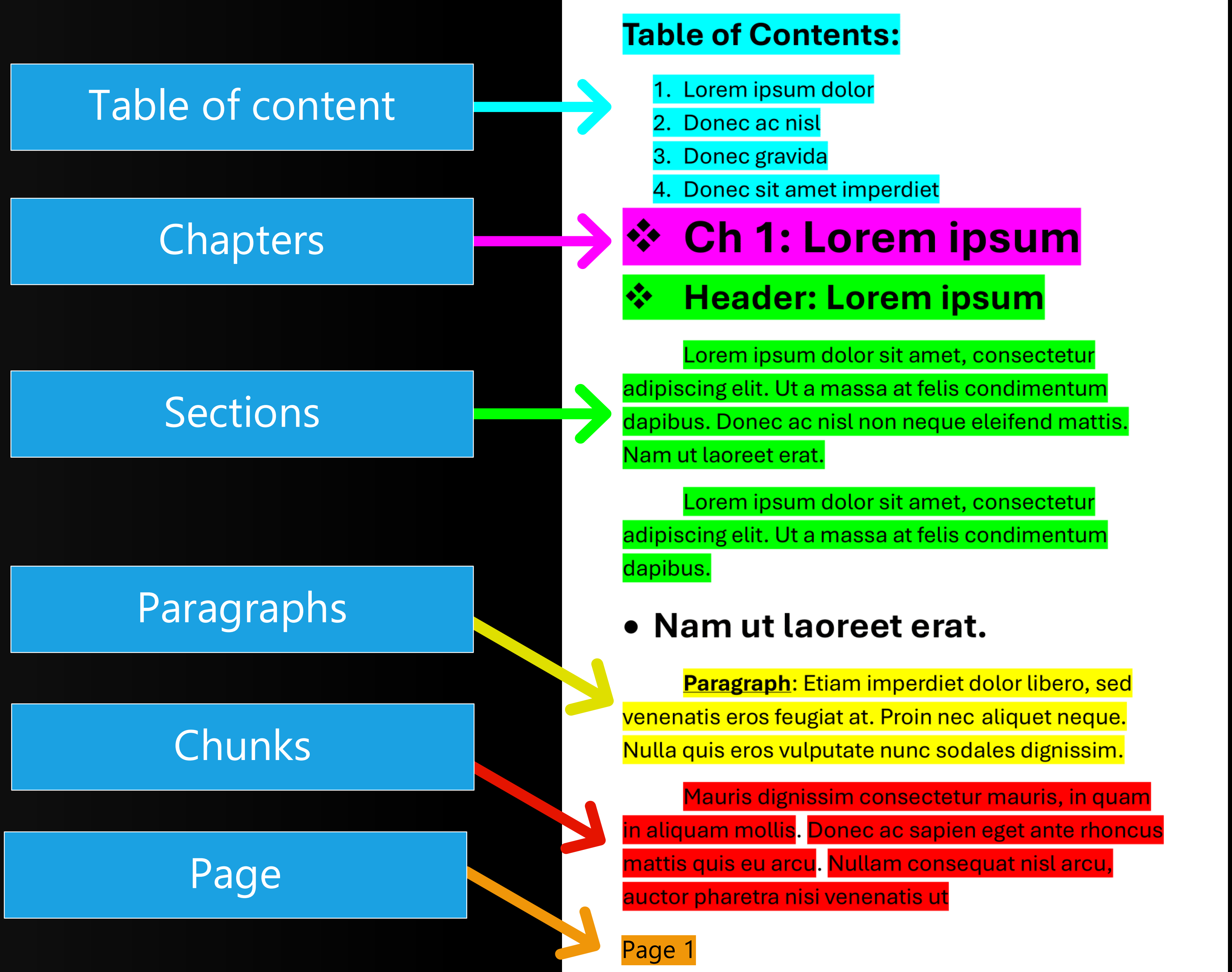


Table of content

Chapters

Sections

Paragraphs

Chunks

Page

Table of Contents:

1. Lorem ipsum dolor
2. Donec ac nisl
3. Donec gravida
4. Donec sit amet imperdiet

❖ Ch 1: Lorem ipsum

❖ Header: Lorem ipsum

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut a massa at felis condimentum dapibus. Donec ac nisl non neque eleifend mattis. Nam ut laoreet erat.

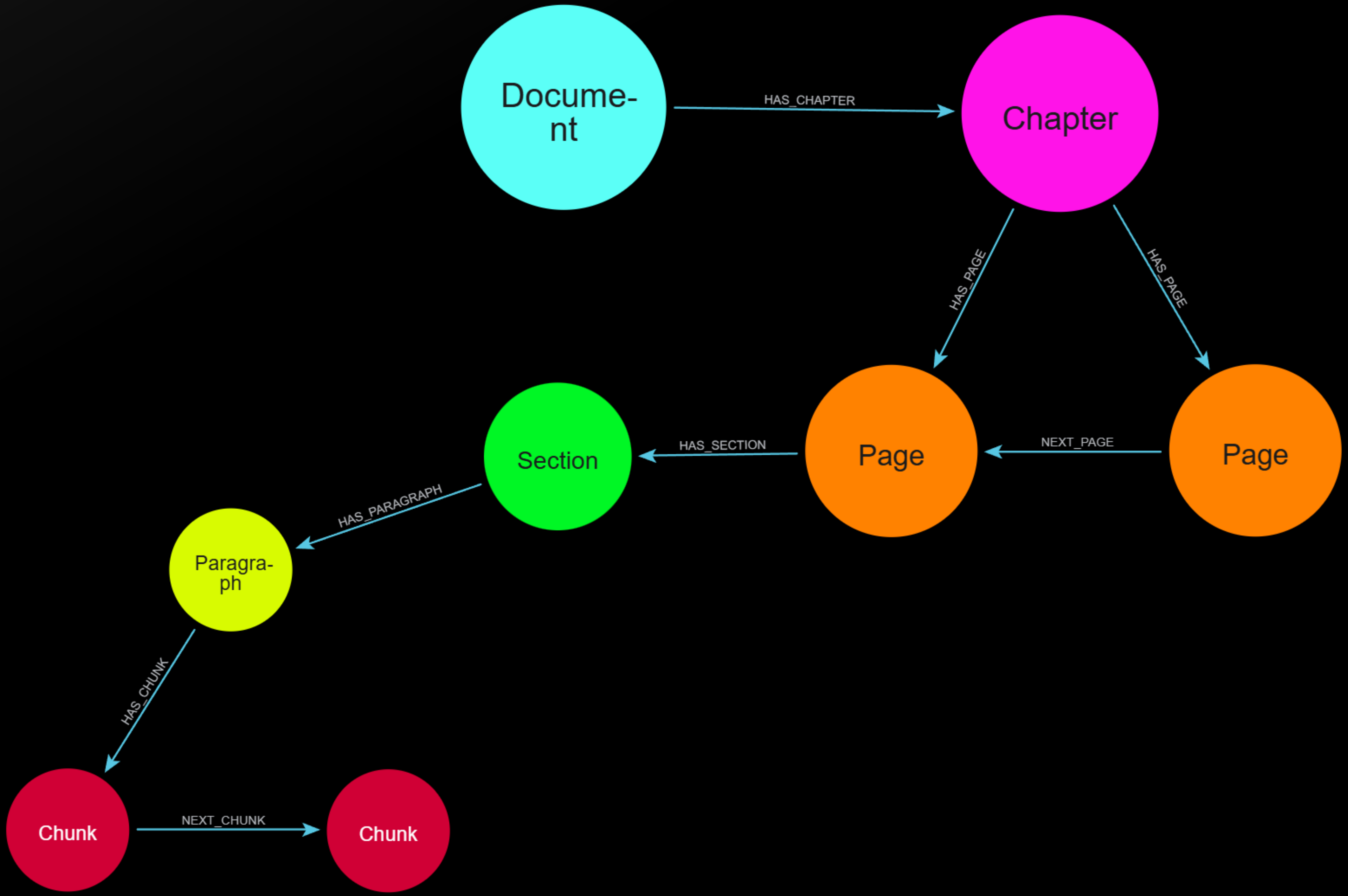
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut a massa at felis condimentum dapibus.

- **Nam ut laoreet erat.**

Paragraph: Etiam imperdiet dolor libero, sed venenatis eros feugiat at. Proin nec aliquet neque. Nulla quis eros vulputate nunc sodales dignissim.

Mauris dignissim consectetur mauris, in quam in aliquam mollis. Donec ac sapien eget ante rhoncus mattis quis eu arcu. Nullam consequat nisl arcu, auctor pharetra nisi venenatis ut

Page 1



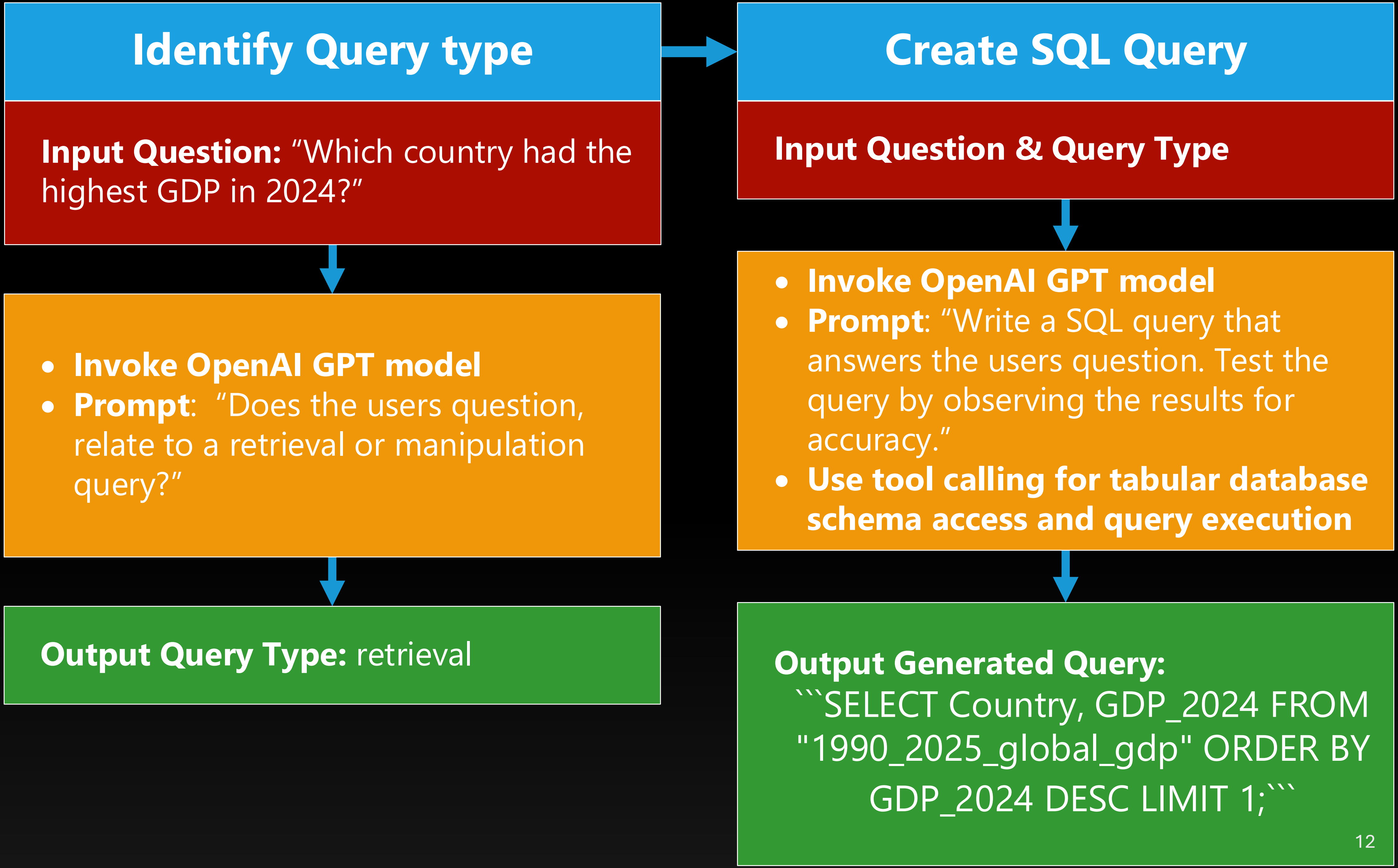
Docume-nt
HAS_CHAPTER
Chapter
HAS_PAGE
HAS_PAGE
Page
NEXT_PAGE
Page
HAS_SECTION
Section
HAS_PARAGRAPH
Paragra-ph
HAS_CHUNK
Chunk
NEXT_CHUNK
Chunk

SQL Agent

The SQL agent interprets the user's question and responds to it with SQL queries that results in an action.

- **Input:** User’s question
- **Output:** Database Query
- **Database:** Reads from tabular database

Input	What Country has the highest happiness index in 2024?
Output	SELECT Country, GDP_2024 FROM "1990_2025_global_gdp" ORDER BY GDP_2024 DESC LIMIT 1;



Identify Query type

Input Question: “Which country had the highest GDP in 2024?”

- **Invoke OpenAI GPT model**
- **Prompt**: “Does the users question, relate to a retrieval or manipulation query?”

Output Query Type: retrieval

Create SQL Query

Input Question & Query Type

- **Invoke OpenAI GPT model**
- **Prompt**: “Write a SQL query that answers the users question. Test the query by observing the results for accuracy.”
- **Use tool calling for tabular database schema access and query execution**

Output Generated Query:

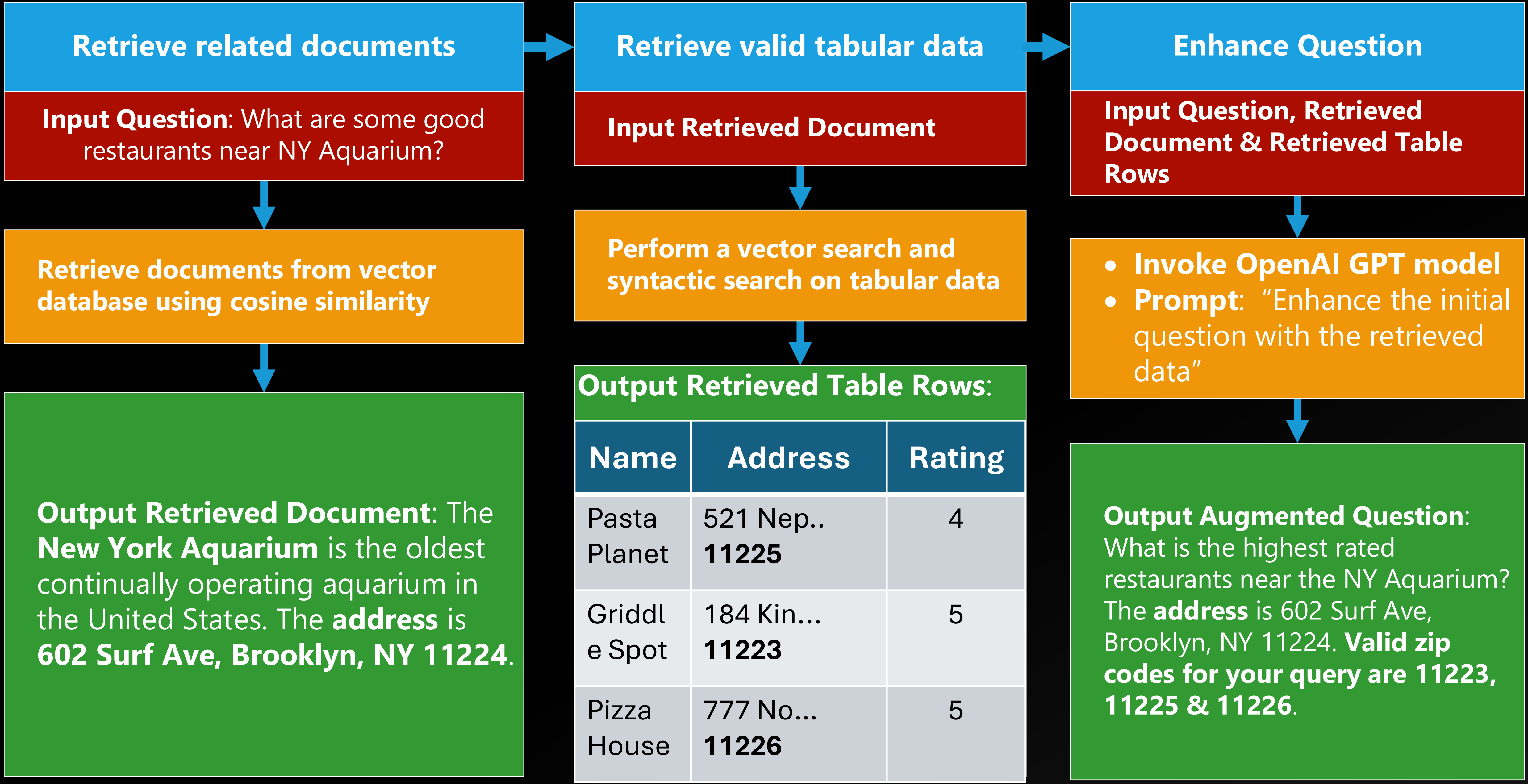
```SELECT Country, GDP_2024 FROM "1990_2025_global_gdp" ORDER BY GDP_2024 DESC LIMIT 1;```

Analyst Agent

Enhances user questions with relevant context from retrieved documents and tabular database before query generation, enabling the SQL Agent to generate precise queries.

- **Input:** User's question
- **Output:** Enhanced version of user's question
- **Database:** Reads from tabular database and document database

Input	What are some good restaurants near NY Aquarium?
Output	What is the highest rated restaurants near the NY Aquarium? The **address** is 602 Surf Ave, Brooklyn, NY 11224. **Valid zip codes for your query are 11223, 11225 & 11226**.



Retrieve related documents

Input Question: What are some good restaurants near NY Aquarium?

Retrieve documents from vector database using cosine similarity

Output Retrieved Document: The **New York Aquarium** is the oldest continually operating aquarium in the United States. The **address** is **602 Surf Ave, Brooklyn, NY 11224**.

Retrieve valid tabular data

Input Retrieved Document

Perform a vector search and syntactic search on tabular data

Output Retrieved Table Rows:

Name	Address	Rating
Pasta Planet	521 Nep.. **11225**	4
Griddle Spot	184 Kin... **11223**	5
Pizza House	777 No... **11226**	5

Enhance Question

Input Question, Retrieved Document & Retrieved Table Rows

- **Invoke OpenAI GPT model**
- **Prompt**: "Enhance the initial question with the retrieved data"

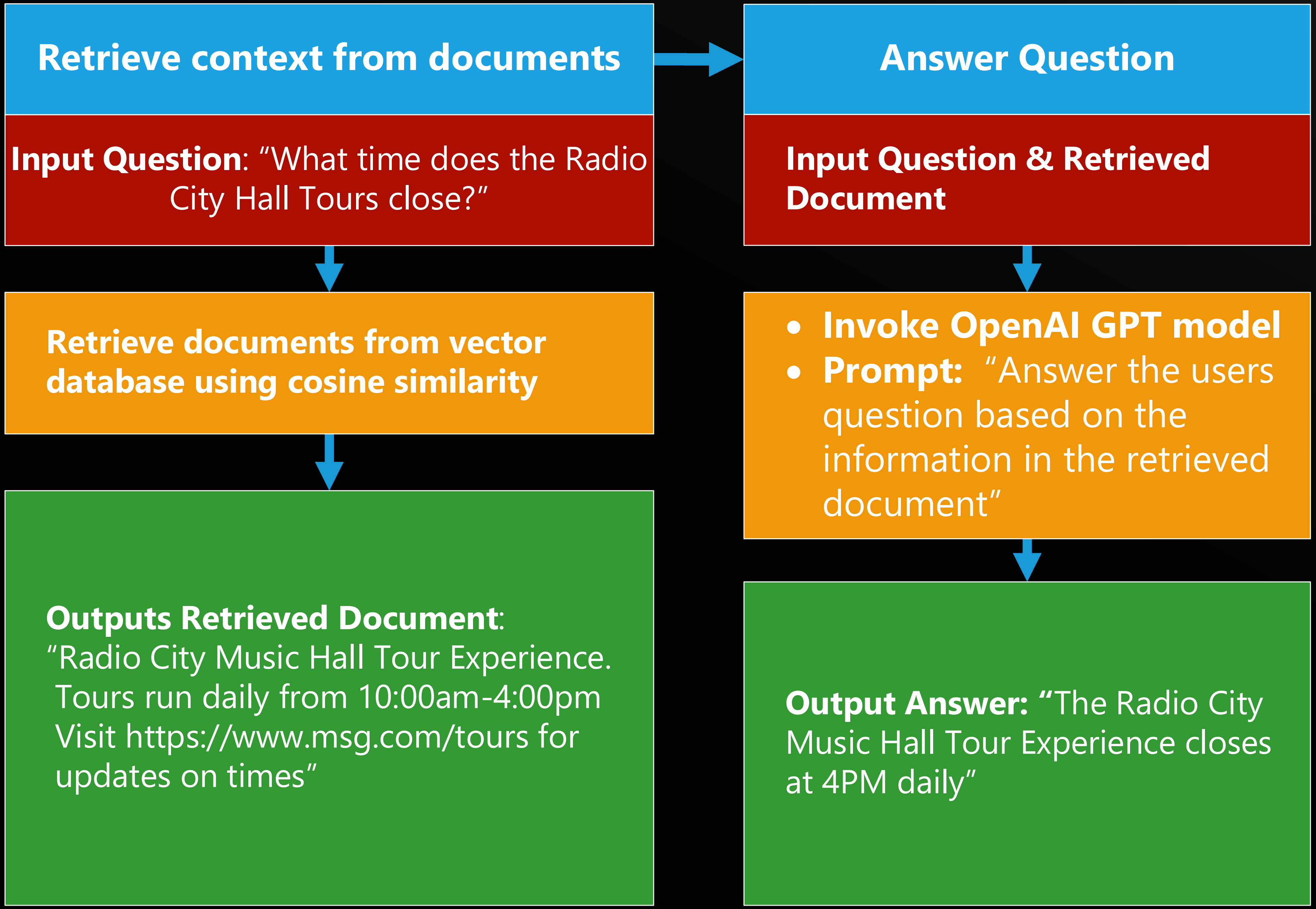
Output Augmented Question: What is the highest rated restaurants near the NY Aquarium? The **address** is 602 Surf Ave, Brooklyn, NY 11224. **Valid zip codes for your query are 11223, 11225 & 11226**.

PDF Agent

Answers user questions directly using retrieved content from documents.

- **Input:** User's question
- **Output:** Answer and retrieved documents
- **Database:** Reads from document vector database

Input	What time does the Radio City Music Hall close?
Output	The Radio City Music Hall Tour Experience closes at 4PM daily



Retrieve context from documents

Input Question: "What time does the Radio City Hall Tours close?"

Retrieve documents from vector database using cosine similarity

Outputs Retrieved Document: "Radio City Music Hall Tour Experience. Tours run daily from 10:00am-4:00pm Visit https://www.msg.com/tours for updates on times"

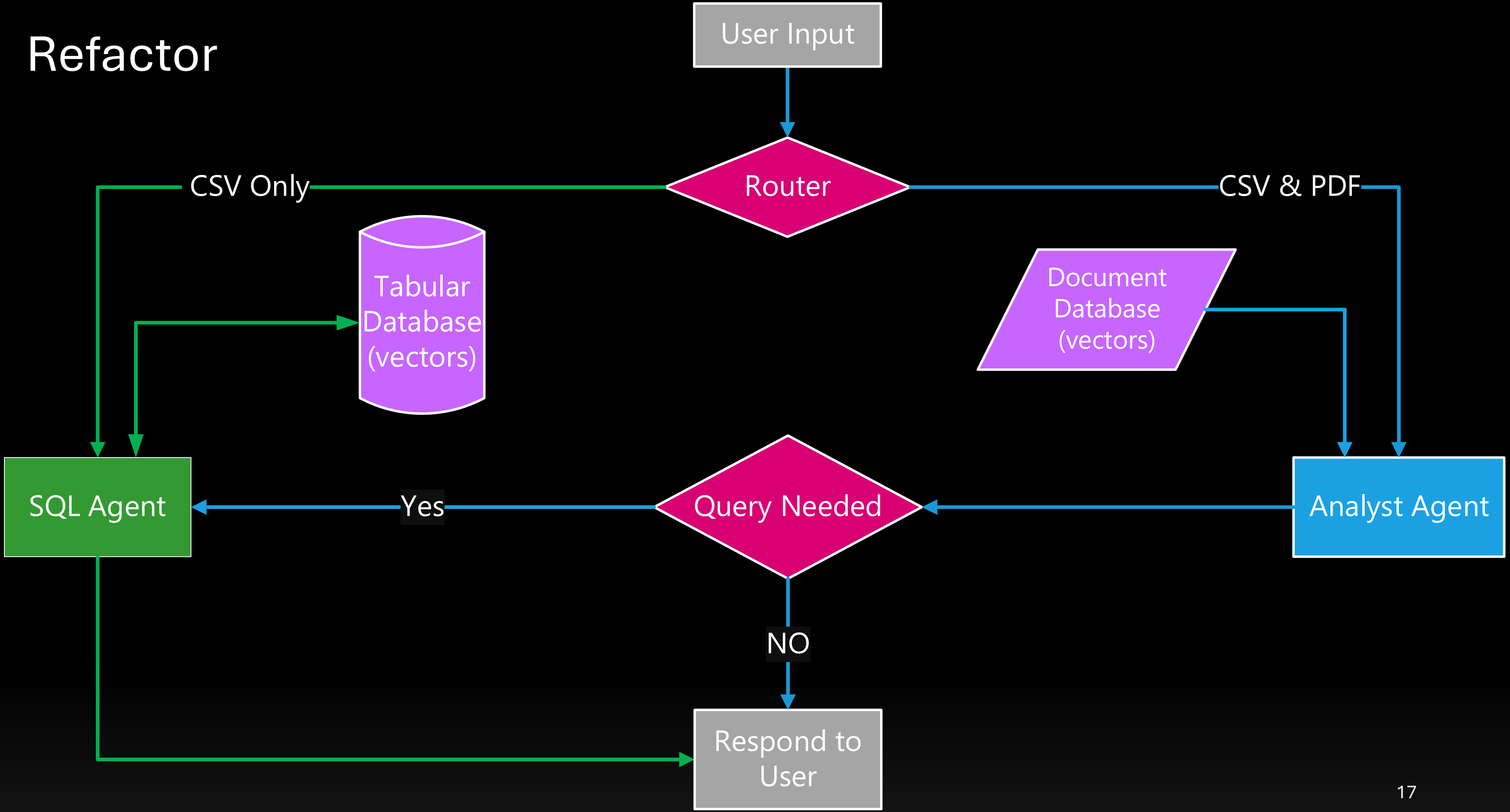
Answer Question

Input Question & Retrieved Document

- **Invoke OpenAI GPT model**
- **Prompt:** "Answer the users question based on the information in the retrieved document"

Output Answer: "The Radio City Music Hall Tour Experience closes at 4PM daily"

Refactor


User Input
Router
CSV Only
CSV & PDF
Tabular Database (vectors)
Document Database (vectors)
SQL Agent
Yes
Query Needed
Analyst Agent
NO
Respond to User

Issue, redundant data flows

Issue: Currently, both the Analyst Agent and the SQL Agent access the tabular database, creating redundant data flows. Restricting unstructured document handling to the Analyst Agent and tabular data to the SQL Agent will improve maintainability and assign each agent to its respective data domain.

Considered changes: Reassign responsibility for tabular data validation/retrieval from the Analyst Agent to the SQL Agent. This will result in the below structure.

- Analyst Agent → Document retrieval + Question enrichment
- SQL Agent → Tabular data validation + Query generation

Issue, is PDF agent necessary?

Issue: The current role of the PDF Agent is limited to document-based question answering using retrieved context. Given that the objective of the application is structured query generation enhanced with document context, the PDF Agent introduces unnecessary redundancy

Considered changes: Remove the PDF Agent. Document retrieval and question enrichment responsibilities are already handled by the Analyst Agent.