JIVA (Graph Enterprise Knowledge Explorer)

**Objective:** JIVA aims to create a unified knowledge graph by combining structured and unstructured data sources, leveraging generative AI for query responses.

**Framework Used:** llamaindex

**Models Used:** GPT-4.5 Turbo and Gemini 1.pro

**Graph Database:** Neo4j

**Tool:** NLSQLTable query engine and custom tools

**Agent:** React

**Project Steps:**

**1.Database Loader:**

a. Connect to the database and generate a CSV file for triple generation, comprising five columns: source node, relationship (foreign key), target node, source table name, and target table name.

b. Create standalone tables for relationships where the source and target table names are the same. And sourec node and tarhet node table are same and relation ship as “self”.

**2.Knowledge Graph Generation for Structured Data:** Generate the knowledge graph from the CSV using llamaindex and create an index for it.

**3.Loading of Unstructured Data:** Load the unstructured data by using simply directory reader and chunk it, treating each page as a separate chunk.

**4.Knowledge Graph for Unstructured Data:** Create a knowledge graph for the PDF on top of the structured graph and create an unstructured index for it.

**6.Removal of Dangling Nodes:** Identify and remove dangling nodes (unconnected nodes) and store or display them in a Neo4j graph.

**7.Customer Tools Creation:**

a.**Structure-only Tool**: Utilize NLSQLTable query engine to convert text to sql,it fetch schema and generate answers from structured data. And pripr to using structure index to query the graph data base and will get related nodes which is reposible to answer the question.and we pass this tables in snlsqltable query.

b. **Unstructured-only Tool**: Create a custom tool using the unstructured index to provide answers from unstructured data.

c. **Struct-Join Tool**: Combine structured and unstructured data sources, with 80% of responses from structured data and 20% from unstructured data.

d. **Mix Tool**: Provide answers from both structured and unstructured data sources, with a 50-50 split.

**9.Filtering and Tool Selection:**

Two filters happen here

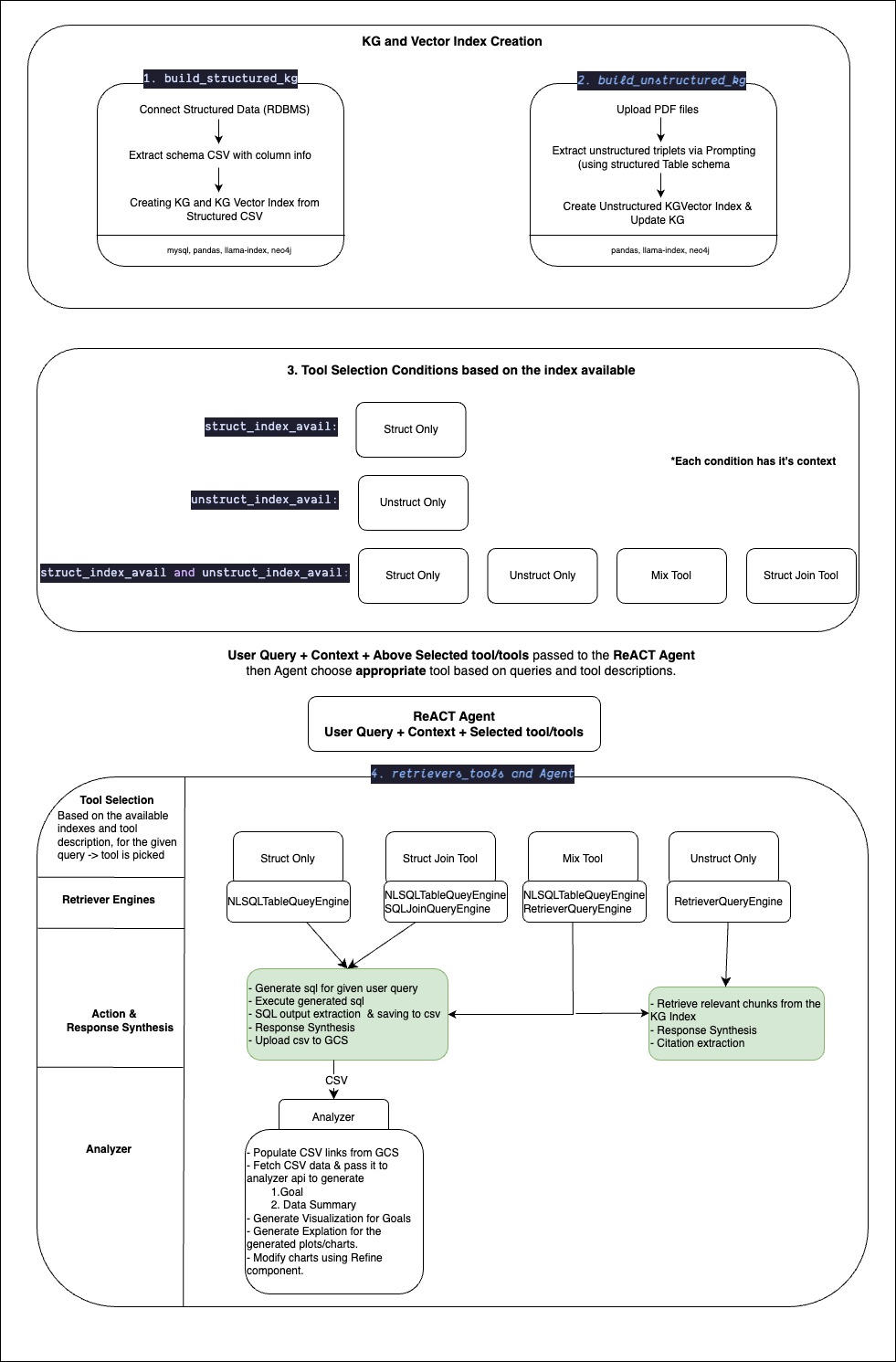
a. Based on available data indexes (structured, unstructured, structured + unstructured), provide the appropriate list of tools to the agent.   
if user uploades only structure data then we pass struct only tool to agent,  
if user uploades only unstructre data then we pass unstruct\_only\_tool to agnet  
if user uploads both then we pass all 4 tools to agent.

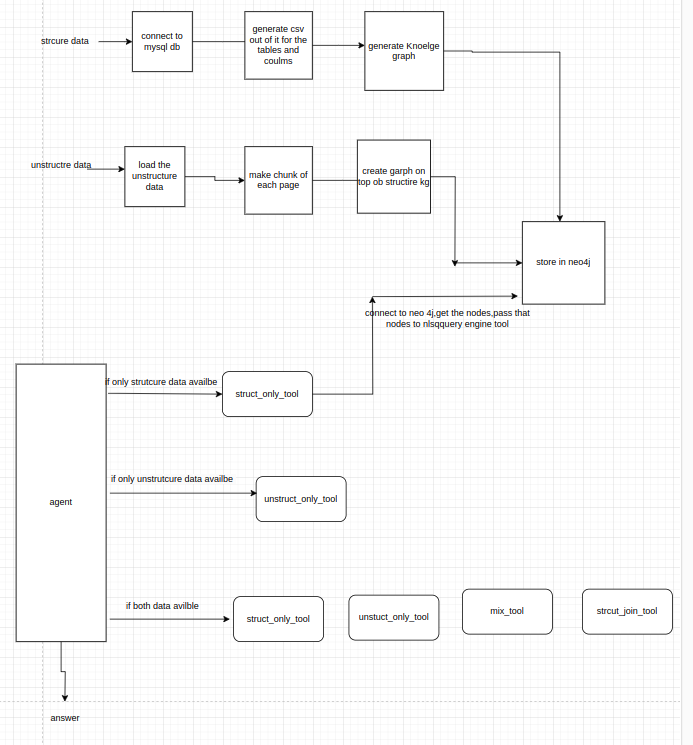
b. The agent (React) selects the appropriate tool based on queries and tool descriptions, using LLM as a reasoning engine.

1. **Description of Struct Tool Creation:** When a structured query is added, fetch related nodes (table names) from the graph database using the structured index and pass them to the NLSQL query engine.
2. TODO : for 3 tools

**Additional Frontend Features:**

1. Display tables for structured data questions generating more than one column in the frontend along with contextual responses.
2. Utilize Microsoft API to generate plots for questions and automatically generate five follow-up questions for each plot.
3. Show citations for unstructured answers.
4. Highlight nodes responsible for generating questions.
5. Provide a gear icon for users to configure parameters such as temperature, depth, model, etc.





Extra:(ignore)

2.struct\_join\_tool---->is not only used for structure its for structure and unstructured (80,20)

3.after creation of unstructure node,we are not updating structure index rather creating unstructure index

++ 4.in langchain we have flow like, tool-->agent ,but in llamaindex ,engine-->tool--->agent

5.you can mention the tools with high clarity that when to use which tool

1. if structure data only ->struct\_only\_tool

2.if unstructured data only --> unstruct\_only\_tool

3.if str+unstr---->unstruct\_only\_tool, struct\_join\_tool, .mix\_tool, struct\_only\_tool

actually two filtering is happening here..if else condition is coming prior to the agent..

1.We are providing the agent with a list of tools based on available data indexes (str,unstr,str+unstr),.

2.and the agent will select the appropriate tool based on queries and tool descriptions, utilizing LLM as a reasoning engine