Retrieving Information from Excel

Solution 1: Using preprocessed dataset as context

Advantages

- Works well for excel files with straightforward structure
- LLM api calls are limited.

Disadvantages

- May not work well for complicated excel sheets.
- Depends on testcase

Solution 2.0 : Neo4j- based Graph Approach

Approach

- Use Neo4j to store and query relationship data extracted from Excel files.

Issue

- Neo4j is a cloud based platform.
- Not accessible given company's security requirements

API Driven Relationship Mapping and Embedding Generation

Approach

- Extract relationship via dedicated API calls.
- Generate graph via NetworkX based on mapping.
- Use embeddings to map user queries to relevant entities in the graph
- Retrieve nearest neighbour nicee from the graph to provide context for response

Outcome

- Successfully maid the queries to related entities.
- High processing time to extract relationships for the entire excel sheet.
- Embedding models have a limited number of tokens, which can lead to problems when Excel cells contain long paragraphs. This results in embeddings misrepresenting the text.

Direct Graph Construction Using NetworkX

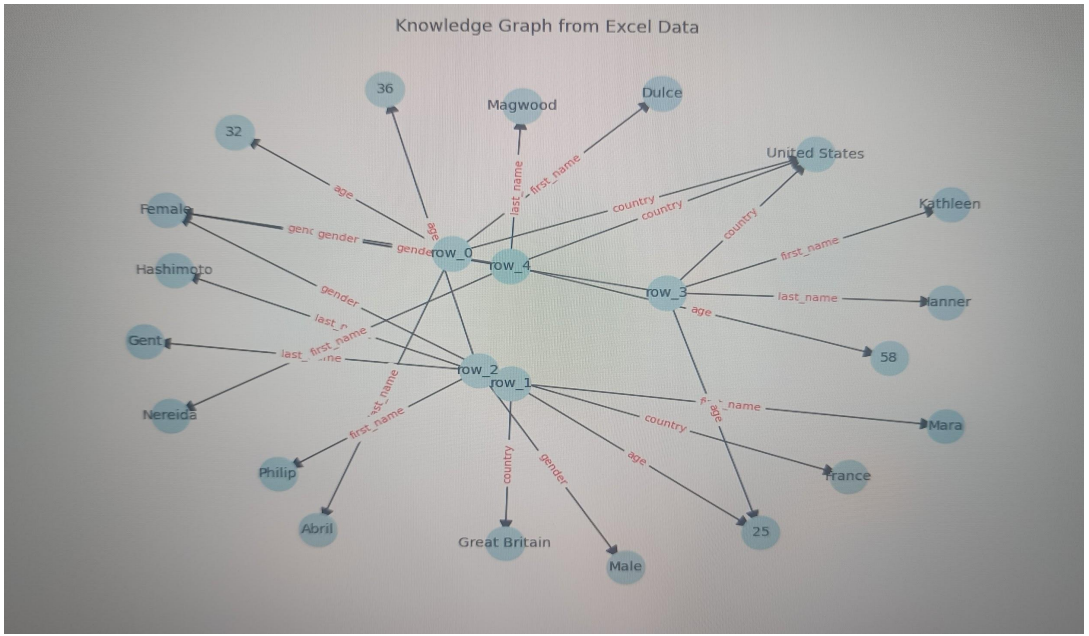
Approach

- Create graphs directly in python via networkx
- Extract entities from queries
- Find relationships wrt to the entities
- Send in those relationships and entities as context.

Outcome

- Successful graph creation and querying in most scenarios

Direct Graph Construction Using NetworkX



Solutions to be Approached

1. Evaluation of alternative libraries like lamaindex or graphrag to handle complex relationships
2. Integrating more advanced chunking and context management techniques
3. Alternative RAG approaches to extract data from excels

Further developments

1. Expanding File Type processing
2. Transition to use In-House applications instead of calling APIs to use Gemini models