Microsoft Fabric Project to Build a Financial Reporting Agent
Project Overview

Overview

The project is designed to leverage the powerful capabilities of **Microsoft Fabric** to streamline and modernize financial data handling and analysis in the wealth management domain. Microsoft Fabric's unified platform offers tools for centralizing diverse financial datasets—ranging from client portfolios and transaction histories —into a centralized **OneLake** data warehouse. This consolidation not only simplifies data management but also ensures seamless access to structured and unstructured data through its versatile **lakehouse and warehouse architecture**. Using **Dataflow Gen2**, the project automates the transfer of data from disparate sources into Fabric's data warehouse, enabling scalable and efficient storage.

The project explains how financial institutions and wealth management firms manage, analyze, and leverage their data for decision-making. By utilizing Microsoft Fabric, the project consolidates diverse financial data sources—such as wealth assets, client portfolios, account transactions —into a centralized data warehouse, ensuring data is easily accessible, secure, and well-organized.

Through the integration of text-to-SQL capabilities powered by Vanna AI, users can perform financial queries using simple natural language, making data analysis more intuitive and accessible even for non-technical users. An interactive reporting dashboard further enhances the experience by offering real-time analytics and visualizations of portfolio performance, asset distribution, transaction histories, and projections.

This approach addresses critical challenges in wealth management, such as fragmented data systems, inefficiencies in report generation, and limited accessibility of insights for decision-making. With streamlined data flows, real-time insights, and user-friendly query capabilities, this solution empowers wealth advisors to better manage client portfolios, optimize investment strategies, and provide personalized recommendations based on deep financial insights. It also enables clients to better understand their financial health and make informed decisions.

Prerequisite Project: Kindly ensure the completion of the LLM Project for building and fine-tuning a large language model before proceeding with this project.

Prerequisites: Basics of LLMs, Vector Databases, Prompt Engineering, Python and Streamlit, Basics of Azure Cloud

Note:

- Utilizing Azure services for this project may result in charges; it is essential to thoroughly review the Azure documentation to understand the pricing structure and potential costs associated with different resources and usage patterns.

- Utilizing Microsoft Fabric is chargeable, with costs depending on the services and resources used. However, a **free trial of 60 days** is available, allowing users to explore its features without initial charges.

Aim

The primary aim of this project is to utilize **Microsoft Fabric** for consolidating wealth and financial data into a centralized warehouse, enabling seamless data integration, storage, and retrieval. By incorporating **text-to-SQL capabilities** and an interactive reporting dashboard powered by **Vanna AI**, the project seeks to streamline financial data analysis, provide real-time insights, and empower wealth advisors and clients to make informed decisions with ease and efficiency.

Data Description

The dataset includes comprehensive financial information crucial for wealth management:

1. **Wealth Assets Data**: Details of client portfolios, including asset classes such as stocks, bonds, real estate, and alternative investments.
2. **Financial Records**: Transaction histories, and investment portfolios for a complete financial overview.

Tech Stack

- ➔ **Language**: Python 3.10
- ➔ **Libraries**: pandas, numpy, Azure, langchain, Open AI, Vanna AI
- ➔ **Cloud Platform Components**: Microsoft Fabric (OneLake, LakeHouse, Fabrics Datawarehouse)
- ➔ **Model**: Retrieval-Augmented Generation (RAG) with GPT-4

➔ **Cloud Platform**: Azure
➔ **Dashboard Framework**: Vanna AI integrated with Microsoft Fabric

Approach

- **Set Up the Microsoft Fabric Environment**
- **Data Preparation and Consolidation**
    - Use **Dataflow Gen2** to copy and import data from the database into Microsoft Fabric
    - Organize data into a format within **OneLake** (lakehouse and warehouse architecture)
- **Data Storage in Microsoft Fabric**
    - Create views and tables in the data warehouse to represent structured data (e.g., portfolio summaries, transaction details)
    - Link the data lakehouse and warehouse for data querying
- **Integration with SQL and Python**
    - Set up an SQL connection string for querying data from Microsoft Fabric
    - Write Python scripts to connect to the SQL engine and retrieve data using the connection string
- **Build a Retrieval-Augmented Generation (RAG) System**
    - Build a **RAG system** using the data schema from the Fabric warehouse to contextualize large language models (LLMs)
    - Use LLMs (e.g., GPT-4) to enable natural language understanding of financial data
    - Validate data accuracy and consistency across OneLake and the data warehouse
- **Develop a Querying Interface**
    - Implement a text-to-SQL conversion layer using Vanna AI for natural language queries
    - Use Python to process SQL queries, interact with the Fabric warehouse, and retrieve results
    - Visualize query outputs using the Vanna AI dashboard

Modular code overview:

Once you unzip the modular_code.zip file, you can find the following:

```
├─ CreateDataWarehouse
│  ├─ Insert to SQL.py
│  ├─ InsertToSQL.py
│  └─ SQL
│     ├─ create_tables.sql
│     ├─ create_views_sql.sql
│     ├─ Proc.json
│     ├─ stored_procedures.sql
│     ├─ Tables.json
│     └─ Views.json
├─ LangchainFabrics
│  ├─ LangChainFabrics.py
│  └─ TestConnectionFabrics.py
├─ RAGToSQL
│  ├─ chroma.sqlite3
│  ├─ FabricsRAG.py
│  ├─ Helper
│  │  ├─ Credentials.py
│  │  ├─ FabricsConnection.py
│  │  ├─ VannaObject.py
│  │  ├─ __init__.py
│  ├─ InferenceRAG.py
│  ├─ TrainingRAG-Artifact
│  │  ├─ .DS_Store
│  │  ├─ Documentation.txt
│  │  ├─ Proc.json
│  │  ├─ Tables.json
│  │  ├─ training_summary.csv
│  │  └─ Views.json
│  ├─ training_summary.csv
│  ├─ TrainRAG.py
│  └─ VisualizeRAG.py
├─ Readme.md
└─ requirements.txt
```

Here is a brief information on the files:

- The folder is organized into three main subdirectories—**CreateDataWarehouse**, **LangchainFabrics**, and **RAGToSQL**—each encapsulating a core component of the project.
- The **CreateDataWarehouse** directory contains scripts and SQL files (`.sql`, `.py`, `.json`) essential for creating and populating a SQL database or data warehouse.
- The **LangchainFabrics** and **RAGToSQL** folders house scripts for leveraging LangChain Fabrics and implementing Retrieval Augmented Generation (RAG) to SQL functionalities, respectively.
- It includes a `Readme.md` file for project documentation and a `requirements.txt` file listing all necessary dependencies.
  Kindly follow all the instructions for running the code from Readme.md file.

Project Takeaways

1. Understand Microsoft Fabric's ecosystem, including OneLake, lakehouse, and data warehouse, as the foundation for modern data management
2. Learn how Microsoft Fabric unifies data from multiple sources into a centralized platform for seamless storage and retrieval
3. Discover how OneLake simplifies data storage by supporting structured and unstructured data formats, enabling scalability and flexibility
4. Explore lakehouse capabilities for blending data lakes and warehouses, offering the best of both worlds for data analysis and management
5. Gain insights into how Microsoft Fabric's SQL engine enables efficient querying and integration with external tools like Python
6. Discover how Fabric supports real-time data processing to provide up-to-date financial insights in dashboards and applications
7. Understand the concept and implementation of RAG systems for contextualized responses
8. Learn how to integrate LLMs like GPT-4 with structured financial data for natural language-based insights
9. Learn how to use Vanna AI to convert natural language queries into SQL statements

10. Understand how Microsoft Fabric supports end-to-end workflows, from data ingestion to reporting and advanced analytics