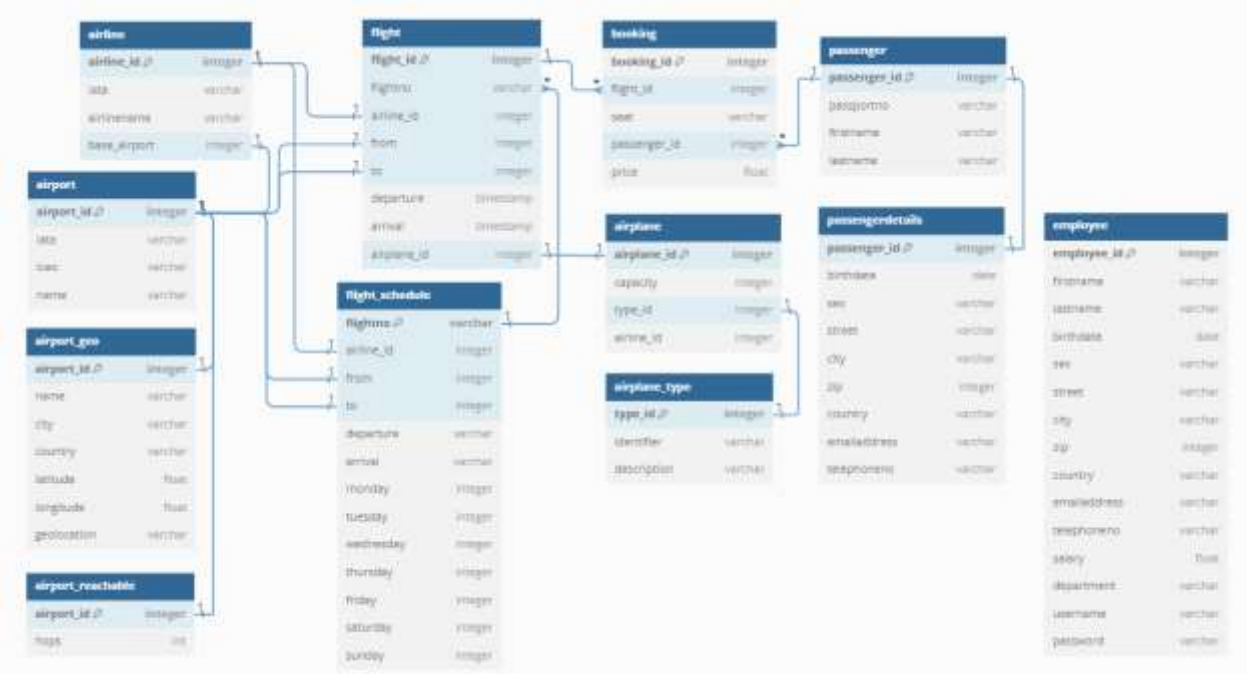
Flughafen Air Travel Analysis

Project by Muhammad Ghulam Abbas and Adnan Ali

Database

We utilized the **Flughafendb** small dataset, a free sample provided by MySQL for demonstrating the warehousing capabilities of MySQL and Oracle products. Although a larger dataset is available, we opted for the smaller dataset, which we minimally modified and primarily retained in its original form. We extracted the data from the MySQL dump into CSV files for our use, focusing on a limited subset of the data.



Data Mart

Objective

To implement a data mart providing analysis of flight bookings.

Context

Our company specializes in flight management. Our transactional database stores comprehensive information about airports, airlines, passengers, flights, and bookings. Notably, a passenger can make multiple bookings on the same flight. The database also contains additional information,

such as details about employees, weather, and flight schedules, which are not pertinent to our analysis.

Star Schema

To structure our data mart, we utilized a star schema for organizing our data, focusing on key dimensions like passengers, flights, and bookings to facilitate efficient querying and reporting.

Data Warehouse Pipeline

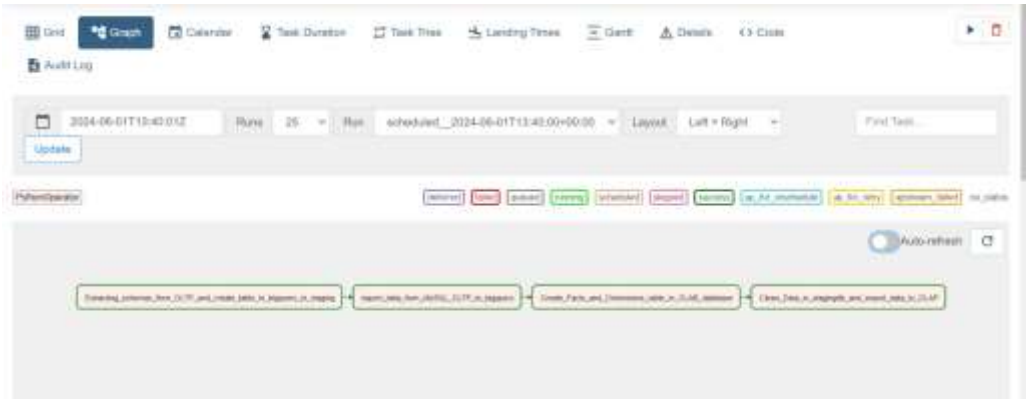
Technologies and Tools

1. **Python**: Orchestrates data manipulation, transformation, and automation, ensuring code reusability and maintainability.
2. **Libraries**: Pandas and Google Cloud Client enhance Python's capabilities for data manipulation and integration with GCP services.
3. **BigQuery**: Serves as the core data repository and processing engine within the GCP environment, enabling storage and querying of vast volumes of structured data securely.

4. **Apache Airflow**: Acts as the orchestrator, providing workflow automation and monitoring functionalities through Directed Acyclic Graphs (DAGs), facilitating efficient execution of tasks.

ETL Process

A "Stored Procedure-Like Approach" encapsulates ETL logic into reusable Python functions, promoting modularity and maintainability. This integrated use of Python, GCP, BigQuery, and Apache Airflow results in a robust and scalable data engineering solution, streamlining ETL processes, and allowing organizations to derive valuable insights from their data assets for informed decision-making and business growth.



Coding Files

The project's coding files serve various functions, from schema conversion to data synchronization between MySQL and BigQuery. Below is a summary of the main scripts used:

1. **Schema Conversion**: Connects to a MySQL database, retrieves schema information for its tables, converts the schema to a BigQuery-compatible format, and generates alter statements for primary keys if necessary.

- ddl_sql2.py
- create_bigquery_table.py

1. **Data Synchronization**: Automates the synchronization of data between a MySQL database and Google BigQuery, ensuring that the data in BigQuery remains up-to-date with the latest information from MySQL.

- export_data_to_bigquery.py

1. **Schema Definition**: Script for creating and defining the schema for an airline reservation system data warehouse in BigQuery.

- Facts_and_Dimensions.py

1. **Data Transfer**: Automates the process of transferring data from an OLTP dataset to an OLAP dataset in Google BigQuery.

- ddl_sql2.py

1. **Data Analysis**: Analyzes data in the BigQuery OLAP data warehouse by executing predefined SQL queries and presenting the results in a clear and structured format.

- Pipeline Python Script (Dimensional Query Functions).py

1. **Fact Table Snapshot**: Creates a snapshot for the fact table by joining all dimensions and facts together.

- Pipeline Python Script (Generation of Fact Table Snapshot).py

Main DAG Airflow File

This Airflow DAG automates the process of moving data from an OLTP system to an OLAP system using Google Cloud services. Here's how it works:

1. **Setup Default Arguments**: The DAG is initialized with default arguments like owner, start date, retries, and retry delay.
2. **Task Definitions**:

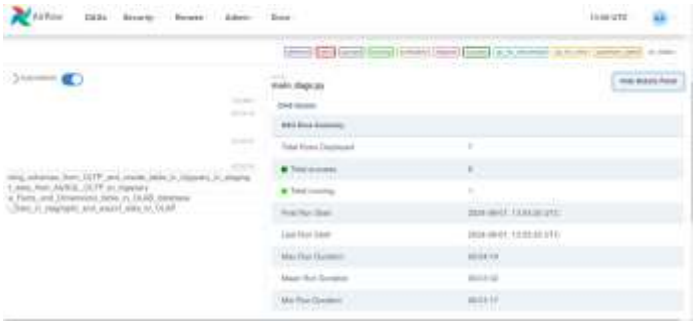
- create_table_in_bigquery: Extracts schemas from the OLTP system and creates tables in BigQuery's staging dataset.
- export_data_to_bigquery: Transfers data from a MySQL OLTP database to BigQuery.
- Facts_and_Dimensions: Creates facts and dimensions tables in the OLAP database.
- export_data_from_OLTP_to_OLAP: Cleans data in the staging database and exports it to the OLAP dataset.

1. **DAG Structure**: Tasks are organized with dependencies using the >> operator, ensuring they run sequentially. For instance, create_table_in_bigquery must finish before export_data_to_bigquery starts.
2. **Schedule Interval**: The DAG is scheduled to run every 10 minutes ("*/10 * * * *").
3. **Catchup and Max Active Runs**: Parameters like catchup and max_active_runs control Airflow's handling of past runs and the maximum number of concurrent DAG runs.
4. **Tags**: Tags are added to categorize the DAG for organizational purposes.

When triggered, Airflow schedules and executes tasks based on dependencies and schedule intervals. Each task invokes a Python function to perform its data processing task. Airflow logs the status and results of each task, providing visibility into the pipeline's progress and any issues

encountered. This setup ensures a robust and automated data pipeline, facilitating efficient data movement from OLTP to OLAP systems.

- main_dags.py



Power BI Integration

The data warehouse is visualized and analyzed using Power BI, with files provided to show the state before and after changes:

- **Power BI File Before Change**: Contains initial visualizations and reports based on the raw dataset.

- **Power BI File After Change**: Updated to reflect the transformations and data integrity checks performed during the ETL process.

Diagram for Data Warehouse

A detailed diagram illustrating the data flow and schema design within the data warehouse, providing a visual overview of the ETL process and the relationships between different data entities.

Video Presentation

A comprehensive video presentation explaining the project, the data warehouse design, the ETL process, and the insights derived from the data analysis.

By integrating all these components, we have developed a robust data warehouse solution that facilitates efficient analysis of flight bookings, providing valuable insights to support business decisions.

https://www.loom.com/share/4d056c0d38604e3b81946fc612f0c7cd?sid=f37663b1-270a-4987-b3b6-2bef157adb23