Airline Data ETL Pipeline Report

Contents

[**Overview** 1](#_Toc181968303)

[**Features** 1](#_Toc181968304)

[**Project Structure** 2](#_Toc181968305)

[1. etl\_scripts 2](#_Toc181968306)

[2. raw\_data\_scripts 2](#_Toc181968307)

[3.Data 2](#_Toc181968308)

[4. analytics 2](#_Toc181968309)

[**Prerequisites** 3](#_Toc181968310)

[**Getting Started (run the pipeline)** 3](#_Toc181968311)

**Overview**

* The Airline Data ETL (Extract, Transform, Load) Pipeline is designed to the collection, processing, and analysis the OpenFlights dataset.
* The pipeline uses PostgreSQL with geospatial extensions (PostGIS) to manage and query vast datasets effectively.
* The pipeline has a graphical user interface (GUI) built with **Tkinter** that allows users to interact with and run the steps of the ETL process and there are Pipeline without GUI

**Features**

The ETL pipeline encompasses several key components:

• Data Ingestion: Downloads relevant datasets (airports, airlines, and routes) from the OpenFlights.

• Data Storage: The raw data is stored in a PostgreSQL database without any processing

• Data Transformation: a series of transformations to clean, normalize, and make it suitable for analysis.

• Data Enrichment: The pipeline calculates direct flight distances, identifies codeshare flights, and appends additional geographic metadata to the data.

• Data Load: After transformation, the enriched data is loaded into a **data warehouse** schema in PostgreSQL for efficient querying and reporting.

• Data Querying: SQL queries are provided to answer specific business questions.

• Business Intelligence: generates insights, such as recommendations for more efficient flight routes, environmental impact analysis, and identifying new market opportunities for airlines.

**Project Structure**

The project is organized into several directories, each serving a specific purpose:

### 1. etl\_scripts

Contains scripts for the extraction, transformation, and loading processes.

Files:  
 - extract.py: Script for extracting data from source repositories.  
 -transform.py: Data cleaning and transformation operations.  
 - load.py: Loads data into PostgreSQL.  
 - create\_schema.sql: Database schema setup script.  
 - etl.py: Script to run the entire ETL pipeline without GUI.  
 - etl\_gui.py: Script to run the ETL pipeline with a GUI for easier interaction.  
 - requirements.txt: Lists Python dependencies required for the ETL process.  
 - .env: Configuration file storing database credentials and settings.

### 2. raw\_data\_scripts

Dedicated scripts for loading raw datasets into the database.

Files:  
 - load\_raw\_data.py: Script for loading raw data into the PostgreSQL database.  
 - requirements.txt: Lists dependencies needed for raw data loading.  
 -.env`: Configuration for database connection details.

### 3.Data

Contains directories for raw and transformed data.

Subdirectories:  
 - raw\_data: Folder containing raw data in CSV format.  
 - transformed\_data: Folder containing processed data ready for analysis.

### 4. analytics

Contains resources related to data analysis and reporting.

Files:  
 - dashboard.pbix: Power BI dashboard with interactive visualizations.  
 - bi\_report.pdf: Business Intelligence report summarizing insights.  
 - queries.sql: SQL queries used to analyze the data.  
 - results.pdf: PDF file summarizing the query results.  
 - etl\_pipeline\_documentation.pdf: Detailed documentation of the ETL pipeline.  
 - demo.pdf: Demo showcasing the pipeline with UI images of the dashboard and sample queries.

-queries.sql: SQL script for queries.  
 - bi.sql: SQL script for business intelligence-related queries. –

**Prerequisites**

Before running the pipeline, ensure the following dependencies are installed and configured:

• Python (Version 3.6 or higher)

• PostgreSQL Database: For storing and querying data.

• Required Python Libraries: Listed in the `requirements.txt` file.

**Getting Started (run the pipeline)**

Follow these steps to set up and run the ETL pipeline:

1. Clone the Repository  
   Clone the project repository to your local environment:  
   git clone <repository-url>
2. Navigate to the Appropriate Folder  
   Change to the `etl\_scripts` directory:  
   cd etl\_scripts
3. Install Dependencies  
   Install the required Python libraries:  
   pip install -r requirements.txt
4. Configuration  
   - Database Configuration: Edit the .env file located in the root directory to configure the database credentials. Ensure you create a PostgreSQL database and include its name in the .env file.  
   WARNING: Ensure that you update the `.env` file with accurate database credentials to avoid connection errors.
5. Run the ETL Pipeline  
   You can run the pipeline either step by step or as a full process:  
   - With GUI: python etl\_gui.py  
   - Without GUI: python etl.py