[GITHUB REPOSITORY](https://github.com/OmarAttia95/FHIR_ETL_Project/blob/main/airflow_dag_workflow.py)

|  |
| --- |
| 01000741752 |
| Smart Phone |
| Omar123915@hotmail.com |
| Envelope |
| Omar Hossam Attia |
| User |

Airflow Instructions & Documentation

Dear,

This is a Python script that defines an Airflow DAG for an ETL (Extract, Transform, and Load) workflow. Below is a simple breakdown:

**1. Imports:**

sys: Used to add custom paths to the Python environment (for importing modules from specific directories).

airflow: The core Airflow package to create and manage the workflow.

Datetime and timedelta: Used to define start dates and retry logic for tasks.

Python Operator: The operator to run Python functions as tasks in Airflow.

**2. Directory Setup:**

The line sys.path.insert(0, '/home/omarattia/Take\_Home\_Assesment/worflow\_scripts') is used to include the directory where the custom Python modules are stored. This makes it possible to import and use these modules in the DAG.

**3. Importing Functions:**

Data extraction: Handles extracting data from JSON dump files.

Data transformation\_\*: Functions for various transformation steps of the data (e.g., transforming patient information, encounter events, etc.).

Data loading: Function that loads the transformed data into a SQL database.

**4. DAG Definition:**

default\_args: Defines default arguments for the DAG, like the owner, retries, retry delay, and start date.

Dag: The main DAG object that defines the workflow and its properties, like the schedule (@daily).

**5. Tasks:**

Each task in the DAG corresponds to a specific part of the ETL workflow:

Task 1 (Data Extraction): Calls data extraction function to extract data.

Task 2-7 (Data Transformation): Each transformation task handles a different part of the data transformation pipeline (e.g., transforming patient data, diagnosis reports, etc.).

Task 8 (Data Loading): The final task loads the transformed data into a database using data loading.

**6. Task Dependencies:**

The line data\_extraction\_task >> data\_transformation\_one\_task >> ... >> data\_loading\_task sets the order in which tasks will run.

The extraction happens first, followed by each transformation step in sequence, and finally, the loading step runs last.

**How to Run the Code:**

Install Apache Airflow: You need Airflow installed to run the DAG.

Place the Python Script in the DAG Folder: The Python script should be placed in the Airflow DAGs folder ($AIRFLOW\_HOME/dags or your custom DAGs folder). This folder is where Airflow looks for DAGs.

**Run the Airflow Scheduler**: Start the Airflow scheduler to run tasks through bash terminal.

**Run the Airflow Webserver**: To access the Airflow UI and monitor the workflow, run the Airflow webserver. Also through bash terminal

Omar Hossam Attia

Analytics engineer