Mentoring Week 4 - Flights Data Pipeline

Data Orchestrations - Job Preparation Program - Pacmann AI

****

source: [freepik](https://www.freepik.com/free-vector/scene-people-sitting-cafe_4998235.htm#fromView=search&page=1&position=48&uuid=e901a119-3daa-44c8-971c-a150fdc2c6e0&new_detail=true&query=cafe)

# 

## **Task Description**

As a **Data Engineer**, you are tasked with designing and implementing a data pipeline for a simulated flight booking system. Your responsibilities include :

* **Data Extraction**
  + Extract selected tables from the source database on a daily basis (optionally incrementally).
* **Data Loading**
  + Load the extracted data to MinIO (object store) in CSV format.
  + Then ingest it into PostgreSQL staging schema.
* **Data Transformation**
  + Transform the staging data into dimensional and fact tables using SQL scripts.
* **Workflow Orchestration**
  + Orchestrate all of the above using Airflow DAG with well-structured TaskGroups.

## **Objectives**

This exercise is a continuation of the previous one (Exercise 1). In this exercise, you are required to modify the work you completed earlier. The modifications to be made are as follows:

* Initialize all Variables and Connections.
* Convert task declarations using Dynamic Tasks.
* Make the pipeline more flexible by implementing Incremental Mode.
* Implement Skip Exceptions.
* Set up a Slack Notifier.

## **UPDATE !**

In the data sources, use this [data](https://drive.google.com/file/d/12bEO24rsR71P7cSKngTyfAZ44hOJhz5x/view?usp=sharing).

## **Task 1: Variable & Connections (20 Points)**

In this task, you are expected to initialize variables and connections (using the CLI) when Airflow is started.

1. Variable

The variables you need to initialize are as follows:

* **incremental (daily)**
  + Value : “True” atau “False”. Default value is “True”.
  + Explanation:
    - If set to “True”, the pipeline will perform the Extract and Load processes incrementally based on the “updated\_at” column of each table.
    - If set to “False”, the opposite will occur the Extract and Load processes will be executed once for all data.
* **tables\_to\_extract**
  + Value :

| ['aircrafts\_data', 'airports\_data', 'bookings', 'tickets', 'seats', 'flights', 'ticket\_flights', 'boarding\_passes'] |
| --- |

* + Explanation :
    - This variable will be used to create dynamic tasks during the data extraction process.
* **tables\_to\_load**
  + Value :

| {  "aircrafts\_data": "aircraft\_code",  "airports\_data": "airport\_code",  "bookings": "book\_ref",  "tickets": "ticket\_no",  "seats": [  "aircraft\_code",  "seat\_no"  ],  "flights": "flight\_id",  "ticket\_flights": [  "ticket\_no",  "flight\_id"  ],  "boarding\_passes": [  "ticket\_no",  "flight\_id"  ]  } |
| --- |

* + Explanation:
    - This variable will be used to create dynamic tasks during the data loading process to the staging area. The keys of the dictionary represent the table names, and the values represent the primary keys of those tables.

1. Connection

Connection yang perlu anda inisialisasi adalah sebagai berkut :

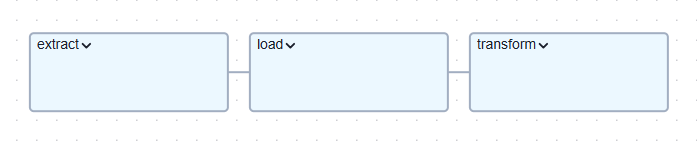
* sources-conn
  + Type : Postgres
  + For the username, password, database, and port, please adjust them according to your data source configuration.
* minio-conn
  + Type : aws
  + For the access\_key, secret\_key, and endpoint\_url, please configure them according to the MinIO setup you have created.
* warehouse-conn
  + Type : Postgres
  + For the username, password, database, and port, please adjust them according to your data warehouse configuration.
* slack\_notifier
  + Type : HTTP
  + Set the WebHook URL according to your actual webhook URL. ***Note: For the WebHook URL, you may replace it with “...” when pushing to GitHub.***

## **Task 2: Dynamic Task (20 Points)**

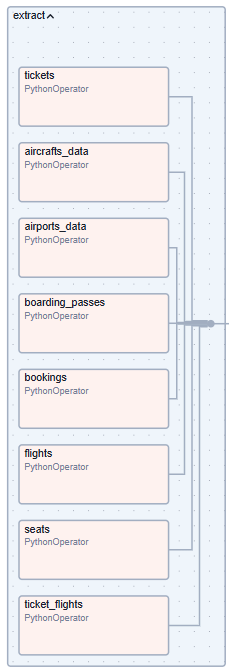
In this task, you are required to replace all task definitions with Dynamic Tasks (Loop Based). You should use the variables you initialized earlier (`tables\_to\_extract` and `tables\_to\_load`).

The result in the Graph View should remain the same as in the previous implementation.

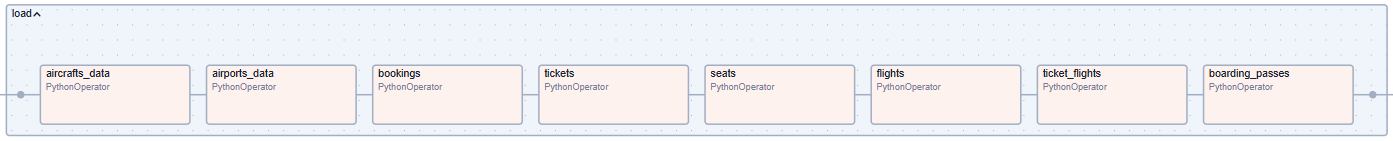
### **All Task Group**



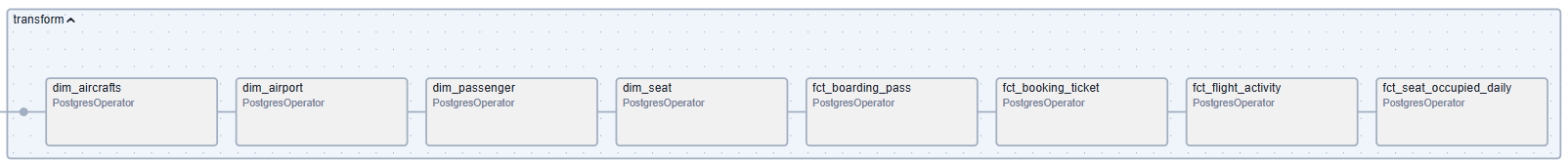
* **Extract Task Group**



* **Load Task Group**



* **Transform Task Group**



* **Modifikasi DAG Details**
* DAG ID : flights\_data\_pipeline
* Schedule : Daily
* Start date : 2025-01-01
* Catchup : True
* Max Active Runs : 1

## **Task 3: Incremental Mode (20 Points)**

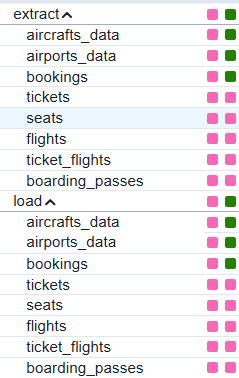
In this task, you are required to modify the **Extract and Load processes** to support daily incremental execution. Whether the process is performed incrementally or not is determined by the **incremental** variable you previously defined.

* If the **incremental** variable is set to **True**, the **Extract and Load processes** should be performed incrementally.
* If it is set to **False**, the processes should perform a full extract and load of all data.

Additionally, you are encouraged to use features such as **Jinja templating** (e.g., {{ ds }}), **XComs**, and other Airflow functionalities to support these modifications effectively.

## **Task 4: Skip Exceptions (20 Points)**

In this task, you are required to modify the **Extract and Load processes** to implement **Skip Exceptions**. The purpose of this task is as follows:



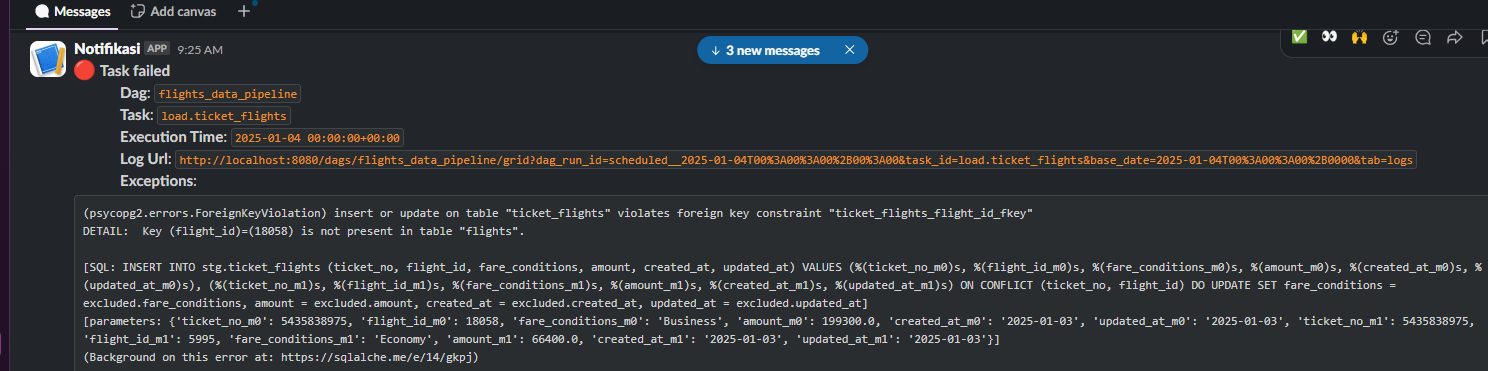
As you can see, the extract processes for the tables bookings, tickets, seats, flights, ticket\_flights, and boarding\_passes are affected by **Skip Exceptions** because there is no data available for the specified date. As a result, the corresponding load processes for these tables are also automatically skipped using Skip Exceptions.

## **Task 5: Slack Notifier (20 Points)**

In this task, you are required to create an **"on failure callback"**, which means that whenever an error occurs or the DAG run fails, a notification will be sent to Slack. The information included in the Slack notification should be as follows:

* **DAG ID**
* **Task ID**
* **Execution Date**
* **Log URL**
* **Exceptions**

Example Result **:**

****

## **Final Output: GitHub Submission**

After completing the exercise, you **must upload your finished project to a GitHub repository**.

### **Your GitHub Repository Must Contain:**

1. All project files and folders
2. A clear and complete **README.md** explaining:
   * Project Overview: What this project does
   * Architecture Description: Diagram or list of components used (Airflow, PostgreSQL, MinIO, etc.)
   * Pipeline Flow : Description of Extract → Load → Transform
   * How to Run and Simulate this project
   * Screenshots of Airflow UI, tasks, etc.

## **Validation Criteria**

Your project will be considered complete if:

* All services run smoothly
* DAG is visible and executable in Airflow
* Data flows from source → MinIO → staging → warehouse
* SQL transformations work properly
* In accordance with the tasks assigned to you.
  + GitHub repo is complete and well-documented