

# **DATA ENGINEERING PROJECT:**

## **BATCH PROCESSING:**

### **OPEN WEATHER ETL DATA PIPELINE ON AWS EC2 INSTANCE USING APACHE-AIRFLOW:**

#### **Project Overview:**

This project involves building an Extract, Transform, and Load (ETL) pipeline using Apache Airflow on an AWS EC2 instance to retrieve weather data from the OpenWeather API and store it in an Amazon S3 bucket. The pipeline is designed to run hourly, extracting current weather data for a specified location (e.g., Islamabad), transforming it into a readable format, and appending the data to CSV files stored in S3. The pipeline uses Airflow to schedule and automate the process.

#### **Key Components:**

1. **OpenWeather API:** Provides the weather data (e.g., temperature, humidity, pressure) for a specified city.
2. **Apache Airflow:** Manages the ETL workflow and scheduling of tasks.
3. **AWS EC2 Instance:** Hosts the Airflow environment.
4. **AWS S3:** Stores the transformed weather data in CSV format.

#### **Steps in the ETL Pipeline:**

1. **Extract:** Retrieve current weather data from the OpenWeather API for the specified city (Islamabad) using an Airflow HTTP sensor and an HTTP operator.
2. **Transform:** Convert temperature from Kelvin to Fahrenheit and format other weather data (humidity, wind speed, pressure, etc.). Add local timestamps for the weather records, sunrise, and sunset times.
3. **Load:** Append the transformed data to a CSV file stored in an S3 bucket. Each new record is added to the existing file, ensuring a continuous, time-series log of weather data.

#### **Prerequisites:**

1. **AWS Account:**
  - Set up an EC2 instance with appropriate security group configurations to allow access to the instance and ensure it has internet access to reach the OpenWeather API.

- IAM Role with S3 access to ensure the EC2 instance can read and write files to an S3 bucket.

## 2. EC2 Instance:

- Choose an appropriate instance type (e.g., t3.medium) for running Airflow.
- Set up the EC2 instance with a compatible operating system (e.g., Ubuntu).

## 3. Apache Airflow:

- Install and configure Apache Airflow on the EC2 instance. Ensure the Airflow web server is accessible for monitoring workflows.
- Use a virtual environment to manage dependencies like boto3, pandas, and pendulum.

## 4. OpenWeather API Key:

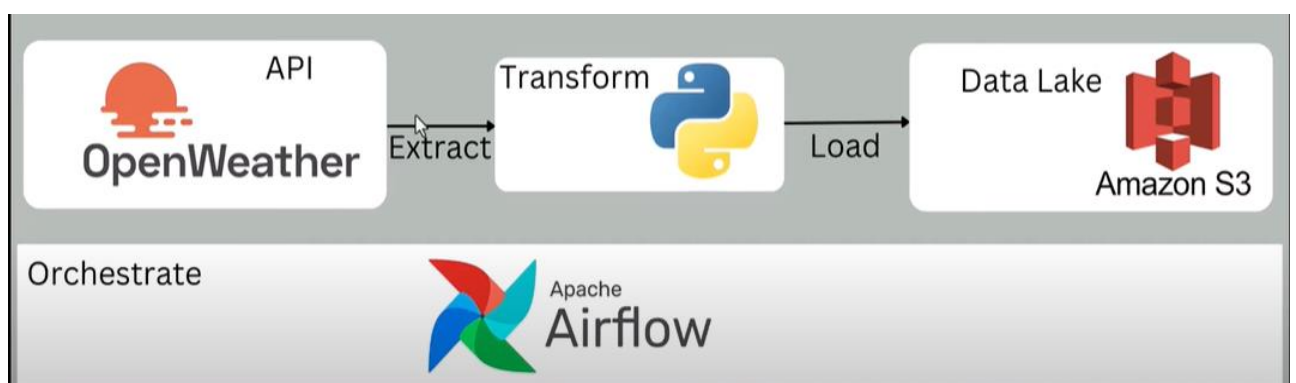
- Sign up on the OpenWeather API website to obtain an API key, which is required to access weather data.

## 5. AWS S3 Bucket:

- Create an S3 bucket where the weather data will be stored. The bucket will contain a CSV file that gets updated with hourly data from the Airflow DAG.

## 6. Airflow HTTP and S3 Connections:

- Set up the HTTP connection in Airflow to interact with the OpenWeather API.
- Configure the AWS S3 connection to enable Airflow to write data to S3.



## INTRODUCTION:

### 1. BASICS OF APACHE-AIRFLOW:

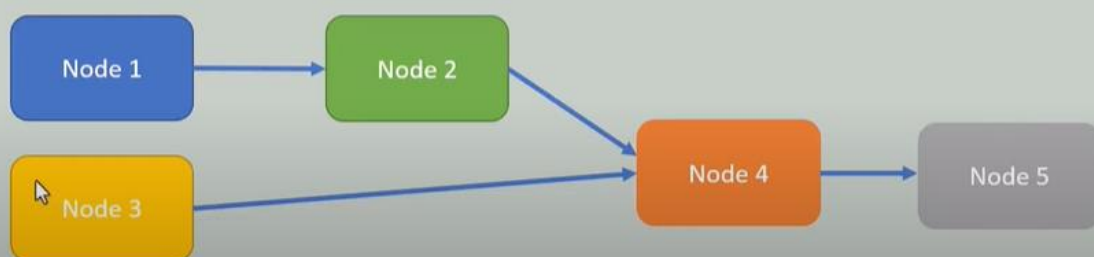
## WHAT IS APACHE AIRFLOW

- ❑ Apache Airflow is an open-source platform used for orchestrating and scheduling workflows of tasks and data pipelines.
- ❑ It provides a way to programmatically author, schedule, and monitor workflows.



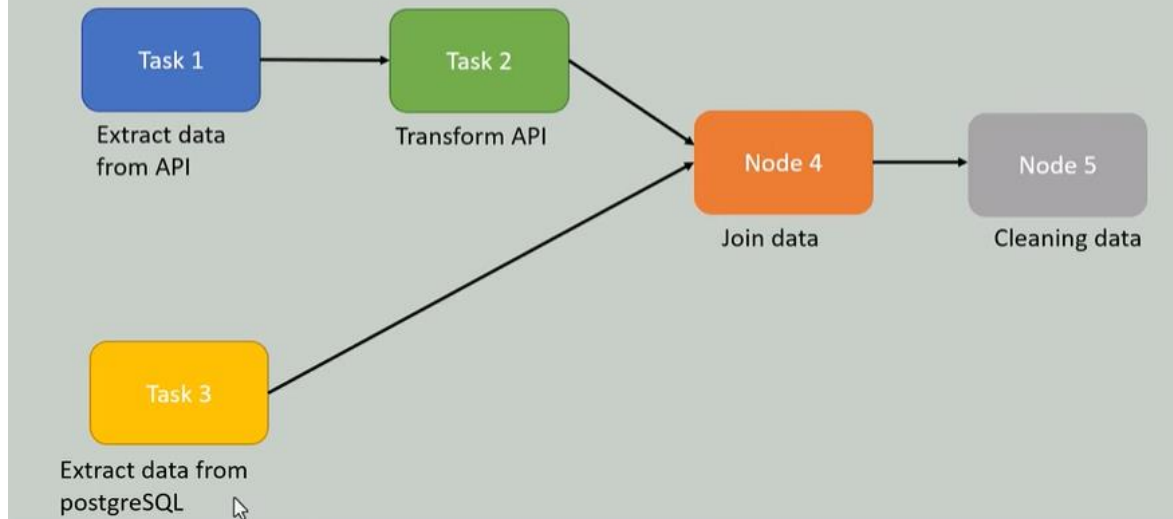
## DAG

- ❑ DAG is Directed Acyclic Graph.
- ❑ It represents collection of tasks and their dependencies.
- ❑ DAG is made of nodes and directed edges.
- ❑ The DAG defines the order in which these tasks should be executed and any dependencies between them.



# OPERATORS

- ❑ The tasks we have in the DAG are called operators.
- ❑ An operator defines a single task in your workflow or data pipeline or DAG.



## STEP NO: 1

### Launching and initializing AWS EC2 Instance:

#### NOTE:

Use t3.medium it is compatible with airflow:

Instances (1/1) Info

Last updated less than a minute ago

Refresh

Connect

Instance state ▼

Actions ▼

Launch Instances

Find Instance by attribute or tag (case-sensitive)

All states ▼

Instance state = running X

Clear filters

< 1 >

\$

<input checked="" type="checkbox"/>	Name <a href="#">🔗</a>	Instance ID	Instance state ▼	Instance type ▼	Status check
<input checked="" type="checkbox"/>	airflow-openweather-instance	i-029b7157a3f9fe907	Running 🔍 🔍	t3.medium	3/3 checks passed

## STEP NO: 2

### Editing the Inbound Rules of EC2 Instance:

### Inbound rules [Info](#)

Security group rule ID	Type <a href="#">Info</a>	Protocol <a href="#">Info</a>	Port range <a href="#">Info</a>	Source <a href="#">Info</a>	Description - optional <a href="#">Info</a>	
sgr-0dc20e106e1f43ab3	HTTP ▼	TCP	80	Cus... ▼	<input type="text" value="0.0.0.0/0"/>	<input type="button" value="Delete"/>
sgr-0b1bafafcb2d16379	SSH ▼	TCP	22	Cus... ▼	<input type="text" value="0.0.0.0/0"/>	<input type="button" value="Delete"/>
sgr-0badf9d351682e915	HTTPS ▼	TCP	443	Cus... ▼	<input type="text" value="0.0.0.0/0"/>	<input type="button" value="Delete"/>
-	All traffic ▼	All	All	An... ▼	<input type="text" value="0.0.0.0/0"/>	<input type="button" value="Delete"/>

## STEP NO: 3

### Installing the required dependencies on EC2 Instance:

- sudo apt update
- sudo apt install python3-pip
- sudo apt install python3.12-venv
- python3 -m venv airflow\_venv *(creating a virtual environment)*

```
ubuntu@ip-172-31-80-120:~$ python3 -m venv airflow_venv
ubuntu@ip-172-31-80-120:~$ ls
airflow_venv
ubuntu@ip-172-31-80-120:~$
```

After creating a virtual environment activate the virtual environment and install the remaining dependencies in that virtual environment:

### Activating the virtual environment:

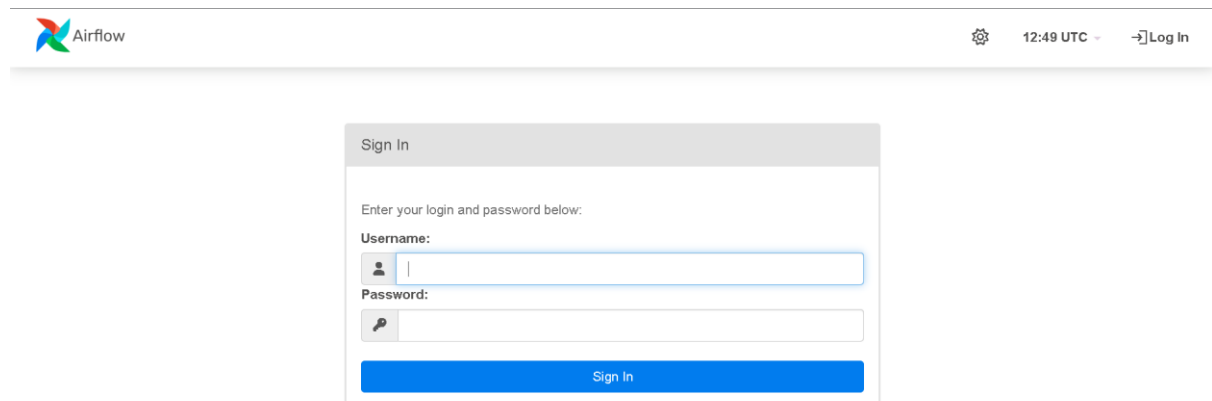
```
ubuntu@ip-172-31-80-120:~$ python3 -m venv airflow_venv
ubuntu@ip-172-31-80-120:~$ ls
airflow_venv
ubuntu@ip-172-31-80-120:~$ source airflow_venv/bin/activate
(airflow_venv) ubuntu@ip-172-31-80-120:~$
```

- pip install pandas
- pip install s3fs
- pip install apache-airflow
- airflow standalone

```
standalone | Airflow is ready
standalone | Login with username: admin password: HGs78VhGeGup3rAX
standalone | Airflow Standalone is for development purposes only. Do not use this in production!
```

**Now that apache-airflow is ready to be used via web browser:**

- Access the airflow on web browser using the following url:
- ec2-54-86-67-70.compute-1.amazonaws.com:8080
- use http instead of https (https will not work)



The screenshot shows the Apache Airflow web interface. At the top left is the Airflow logo. At the top right, there is a settings icon, the time "12:49 UTC", and a "Log In" link. In the center, there is a "Sign In" form. The form has a title "Sign In" and a subtitle "Enter your login and password below:". It contains two input fields: "Username:" and "Password:". The "Username:" field has a user icon on the left. The "Password:" field has a key icon on the left. Below the input fields is a blue "Sign In" button.

## STEP NO: 4

### Adding our connection string in Apache-Airflow:



The screenshot shows the Apache Airflow web interface. At the top left is the Airflow logo. To the right of the logo are several navigation links: "DAGs", "Cluster Activity", "Datasets", "Security", "Browse", "Admin", and "Docs". The "Admin" link is highlighted, and a dropdown menu is open. The dropdown menu contains the following items: "Variables", "Configurations", "Connections", "Plugins", "Providers", "Pools", and "XComs". Below the navigation links, there are two yellow warning boxes. The first box says "Do not use **SQLite** as metadata DB in production – it should only be used for dev/testing. We re... or MySC". The second box says "Do not use the **SequentialExecutor** in production. [Click here](#) for more information."

## DAGs

- Navigate to connections at Apache Airflow:
- Adding a new connection:

Warning: Fields that are currently populated can be modified but cannot be deleted. To delete data from a field, delete the Connection object and create a new one.

### Edit Connection

Connection Id \* weathermap\_api

Connection Type \* HTTP  
Connection Type missing? Make sure you've installed the corresponding Airflow Provider Package.

Description

Host https://api.openweathermap.org

## STEP NO: 5

- Now creating our new folder named as dags inside our airflow folder:
- Mkdir dags

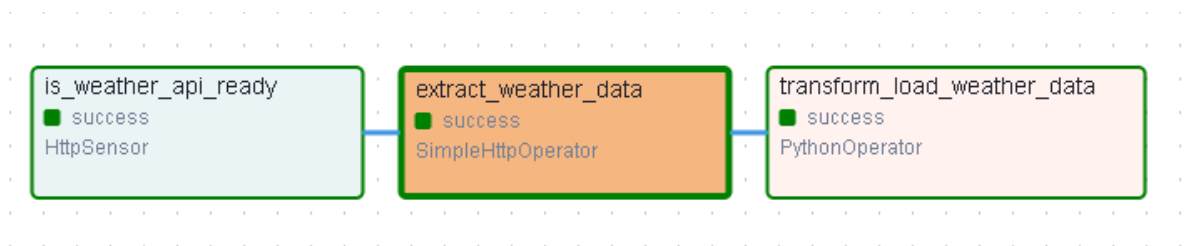
```
ubuntu@ip-172-31-80-120:~/airflow$ ls
airflow-webserver.pid airflow.cfg airflow.db dags logs standalone_admin_password.txt webserver_config.py
ubuntu@ip-172-31-80-120:~/airflow$ cd dags
ubuntu@ip-172-31-80-120:~/airflow/dags$ sudo nano weather_dag.py
```

- Paste the code of weather\_dag.py from visual studio to this ec2 instance file
- Restarting the airflow server and we will see the Dag is being made

☒ weather\_dag airflow @daily 2024-10-01, 00:00:00

## STEP NO: 6

- The tasks are running successfully:



## STEP NO: 7

- Now creating our aws S3 bucket and from EC2 instance give permission To the S3

General purpose buckets (1) <a href="#">Info</a> <a href="#">All AWS Regions</a>			
Buckets are containers for data stored in S3.			
<input type="text" value="Find buckets by name"/> <span>&lt; 1 &gt; ⚙</span>			
Name	AWS Region	IAM Access Analyzer	Creation date
<a href="#">aws-openweather-bucket1</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	October 2, 2024, 15:46:08 (UTC+05:00)

- Navigate to EC2 instance and navigate to security section create a new IAM role:

[IAM](#) > [Roles](#) > Create role

Step 1  
Select trusted entity

Step 2  
Add permissions

Step 3  
Name, review, and create

Select trusted entity [Info](#)

Trusted entity type

☒ **AWS service**  
Allow AWS services like EC2, Lambda, or others to perform actions in this account.

☐ **AWS account**  
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.

☐ **Web identity**  
Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.

☐ **SAML 2.0 federation**  
Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.

☐ **Custom trust policy**  
Create a custom trust policy to enable others to perform actions in this account.

Use case

Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

Service or use case  
EC2

Choose a use case for the specified service.  
Use case  
☒ **EC2**  
Allows EC2 instances to call AWS services on your behalf

## STEP NO: 8

At last, the data is being successfully extracted, transformed and loaded into our S3 bucket after every hour according to our Pakistan's Time Zone!

Objects (3) [Info](#)

[Refresh](#) [Copy S3 URI](#) [Copy URL](#) [Download](#) [Open](#) [Delete](#) [Actions](#) [Create folder](#) [Upload](#)

< 1 > ⚙

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	<a href="#">current_weather_data_Islamabad_021020_24160825.csv.csv</a>	csv	October 2, 2024, 16:03:27 (UTC+05:00)	349.0 B	Standard
<input type="checkbox"/>	<a href="#">current_weather_data_Islamabad_021020_24160512.csv.csv</a>	csv	October 2, 2024, 16:05:13 (UTC+05:00)	349.0 B	Standard
<input type="checkbox"/>	<a href="#">current_weather_data_Islamabad_021020_24160825.csv.csv</a>	csv	October 2, 2024, 16:08:26 (UTC+05:00)	349.0 B	Standard

# THE END