DATA ENGINEERING PROJECT:

BATCH PROCESSING:

OPEN WEATHER ETL DATAPIPELINE 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:

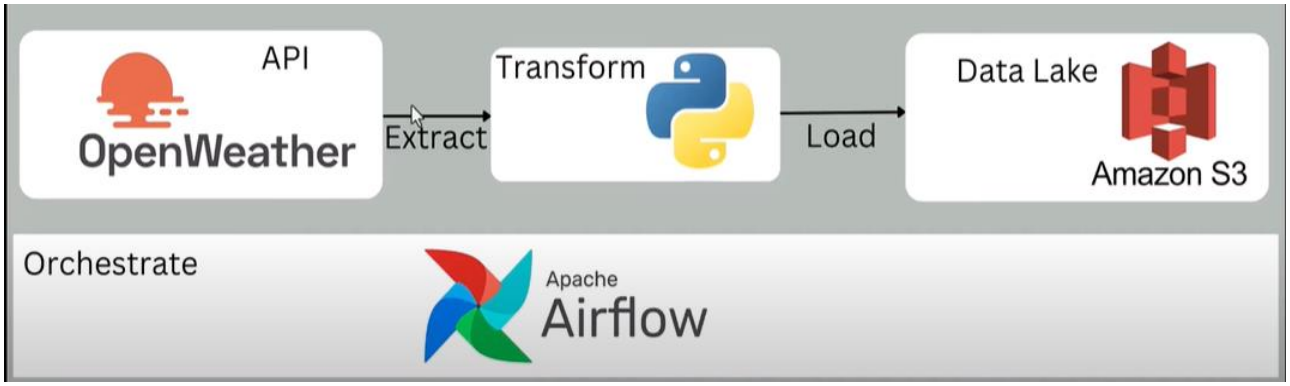
1. Extract: Retrieve current weather data from the OpenWeather API for the specified city (Islamabad) using an Airflow HTTP sensor and an HTTP operator.
2. Transform: Convert temperature from Kelvin to Fahrenheit and format other weather data (humidity, wind speed, pressure, etc.). Add local timestamps 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.

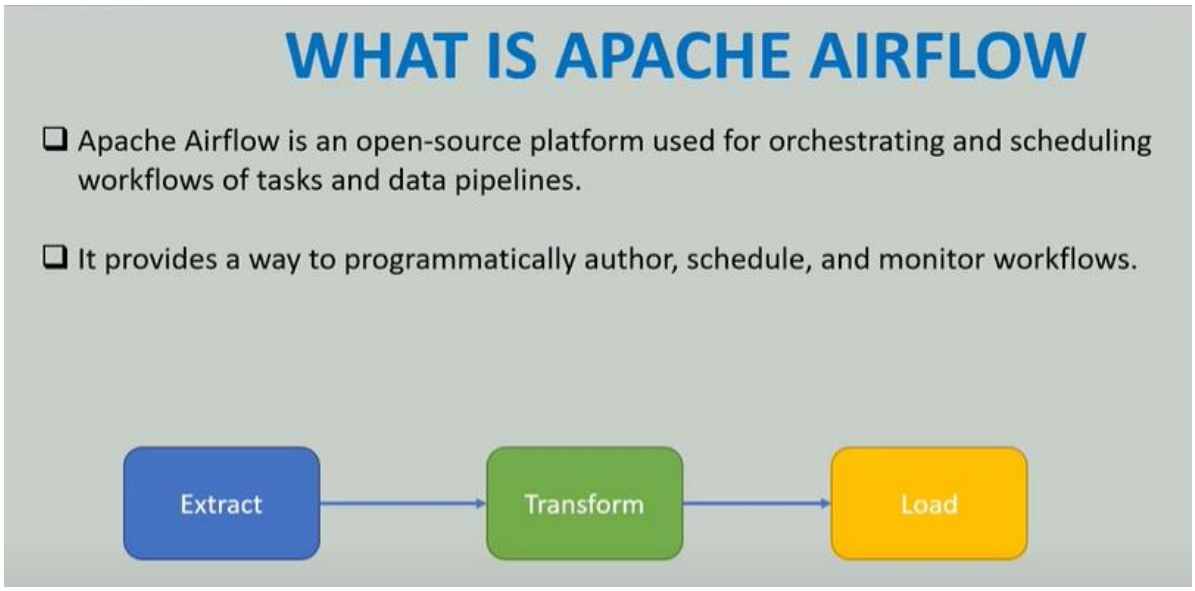
- IAM Role with S3 access to ensure the EC2 instance can read and write files to an S3 bucket.

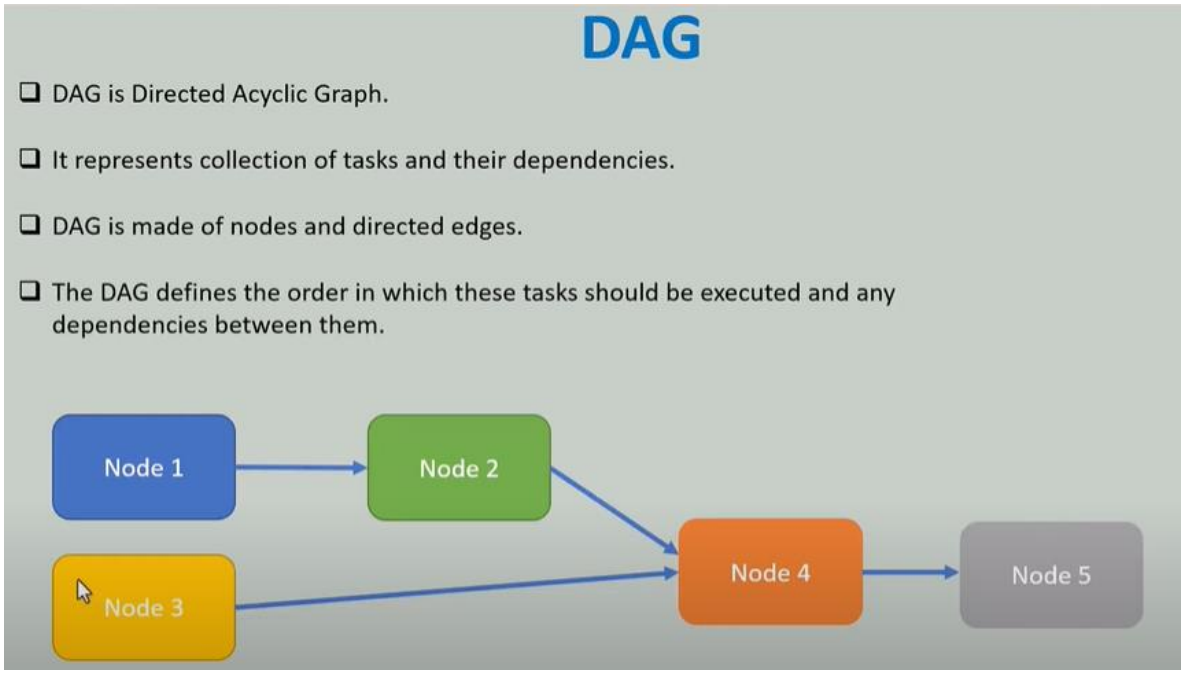
2. **EC2 Instance:**
   - Choose an appropriate instance type (e.g., t3.medium) for running Airflow.
   - Set up the EC2 instance with a compatible operating system (e.g., Ubuntu).
3. **Apache Airflow:**
   - Install and configure Apache Airflow on the EC2 instance. Ensure the Airflow web server is accessible 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.

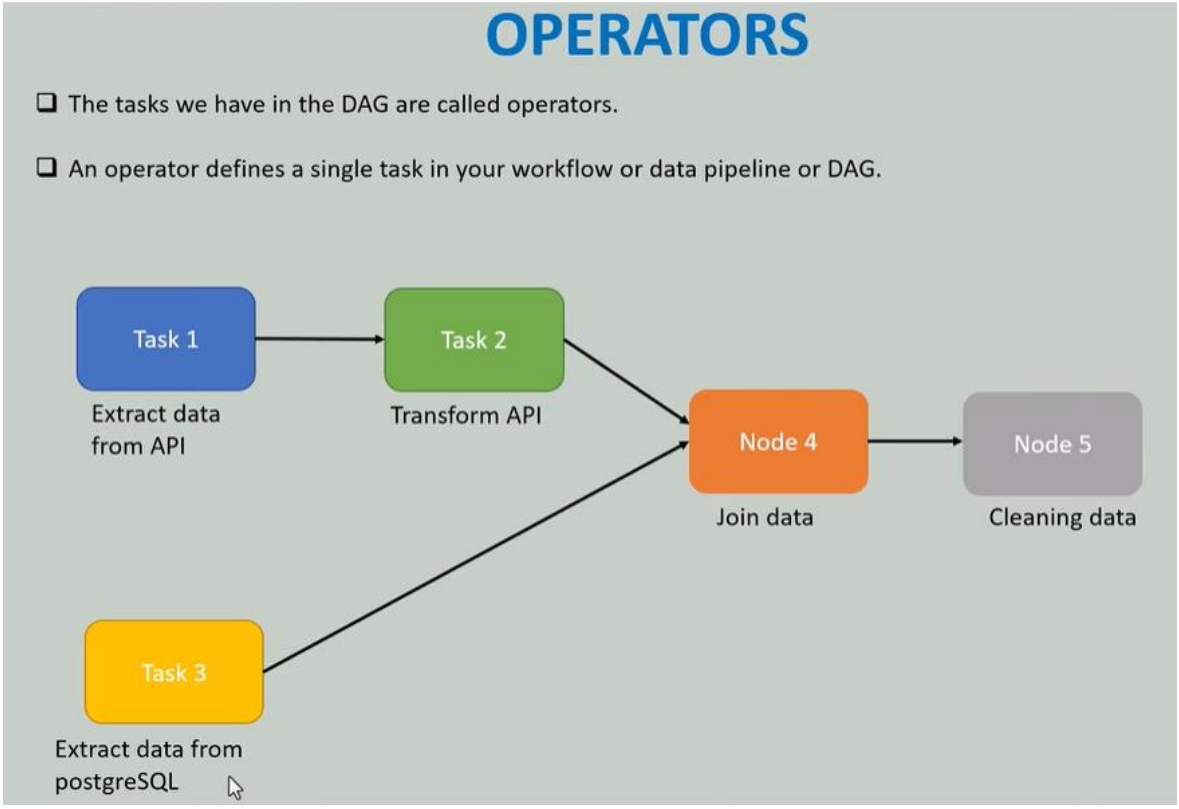


INTRODUCTION:

1. BASICS OF APACHE-AIRFLOW:





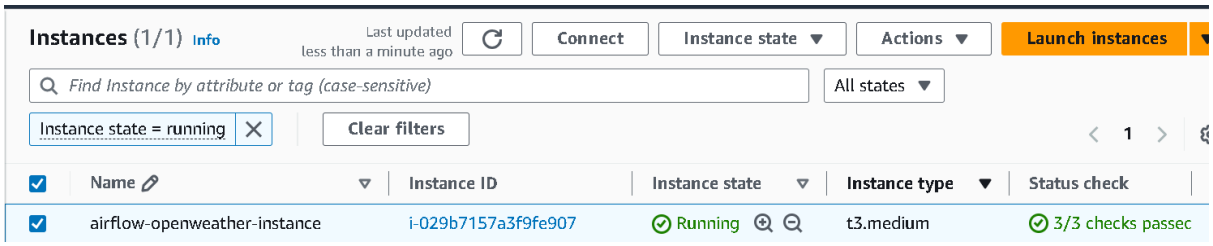


STEP NO: 1

Launching and initializing AWS EC2 Instance:

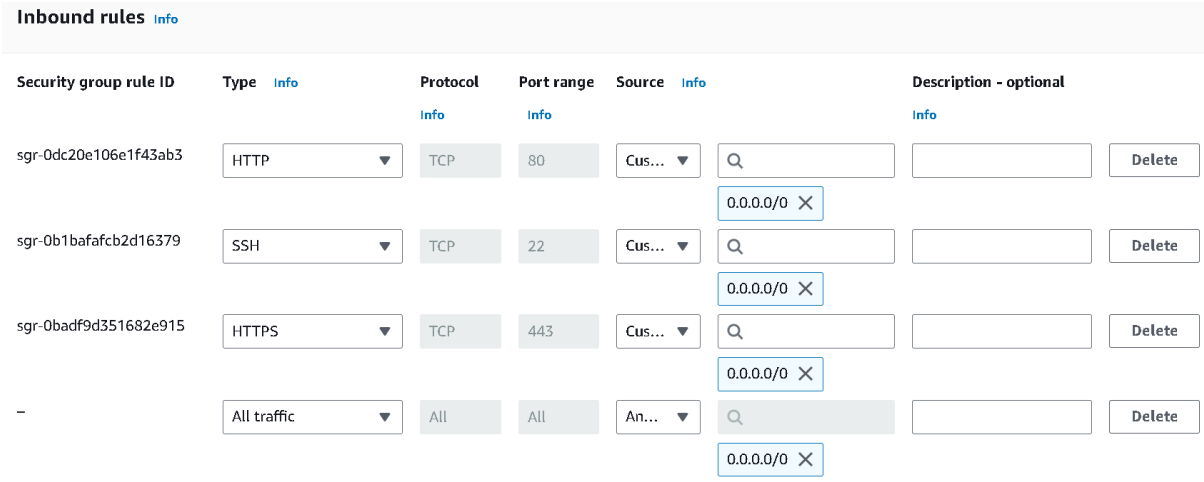
NOTE:

Use t3.medium it is compatible with airflow:



STEP NO: 2

Editing the Inbound Rules of EC2 Instance:



Inbound rules Info

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
sgr-0dc20e106e1f43ab3	HTTP	TCP	80	Cus...	0.0.0.0/0		Delete
sgr-0b1bafafcb2d16379	SSH	TCP	22	Cus...	0.0.0.0/0		Delete
sgr-0badf9d351682e915	HTTPS	TCP	443	Cus...	0.0.0.0/0		Delete
–	All traffic	All	All	An...	0.0.0.0/0		Delete

STEP NO: 3

<u>Installing the required dependencies on EC2 Instance:</u>

- 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:

<u>Activating the virtual environment:</u>

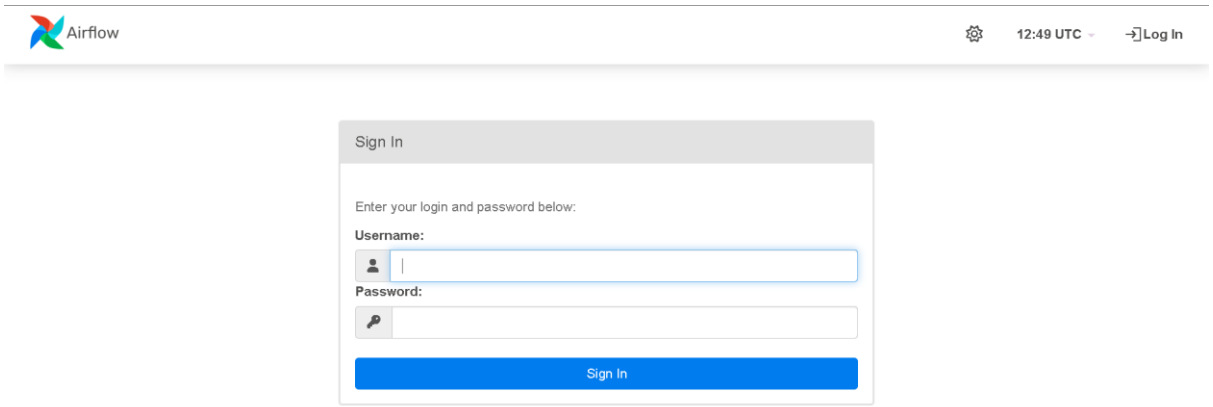
```
ubuntu@ip-172-31-80-120:~$ python3 -m venv airflow_venv
ubuntu@ip-172-31-80-120:~$ ls
airflow_venv
ubuntu@ip-172-31-80-120:~$ source airflow_venv/bin/activate
(airflow_venv) ubuntu@ip-172-31-80-120:~$
```

- 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:

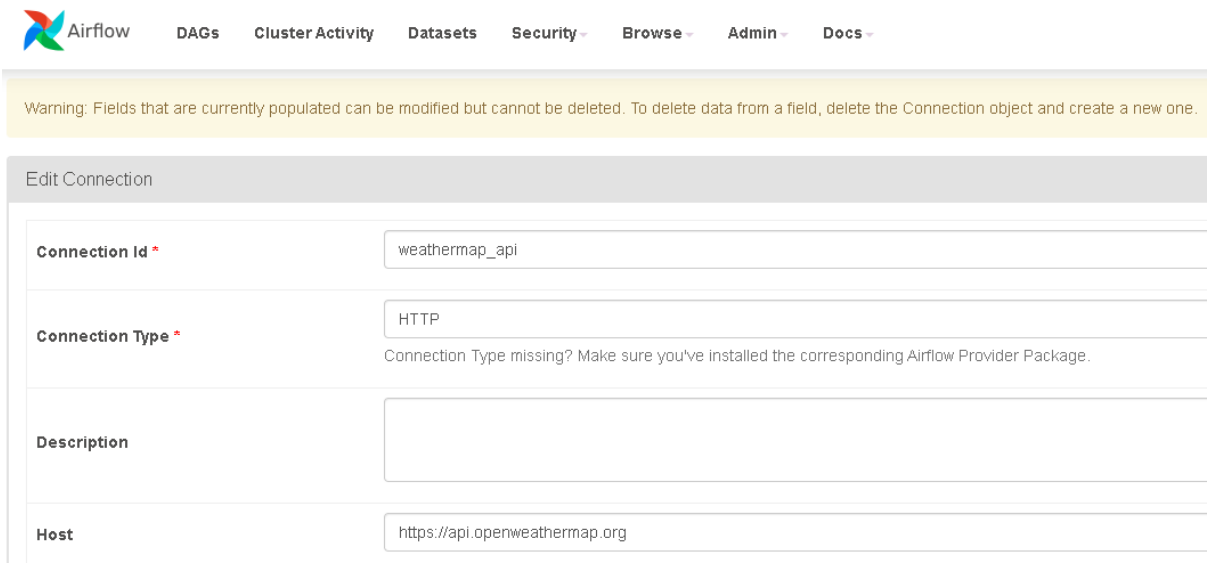
- Access the airflow on web browser using the following url:
- ec2-54-86-67-70.compute-1.amazonaws.com:8080
- use http instead of https (https will not work)



STEP NO: 4

<u>Adding our connection string in Apache-Airflow:</u>

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



STEP NO: 5

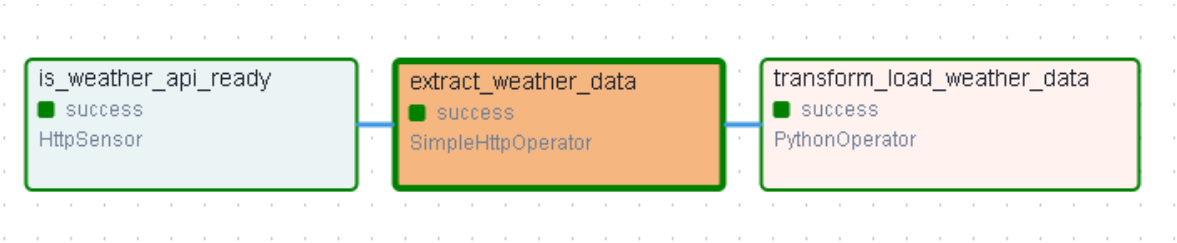
- Now creating our new folder named as dags inside our airflow folder:
- Mkdir dags

```
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

STEP NO: 6

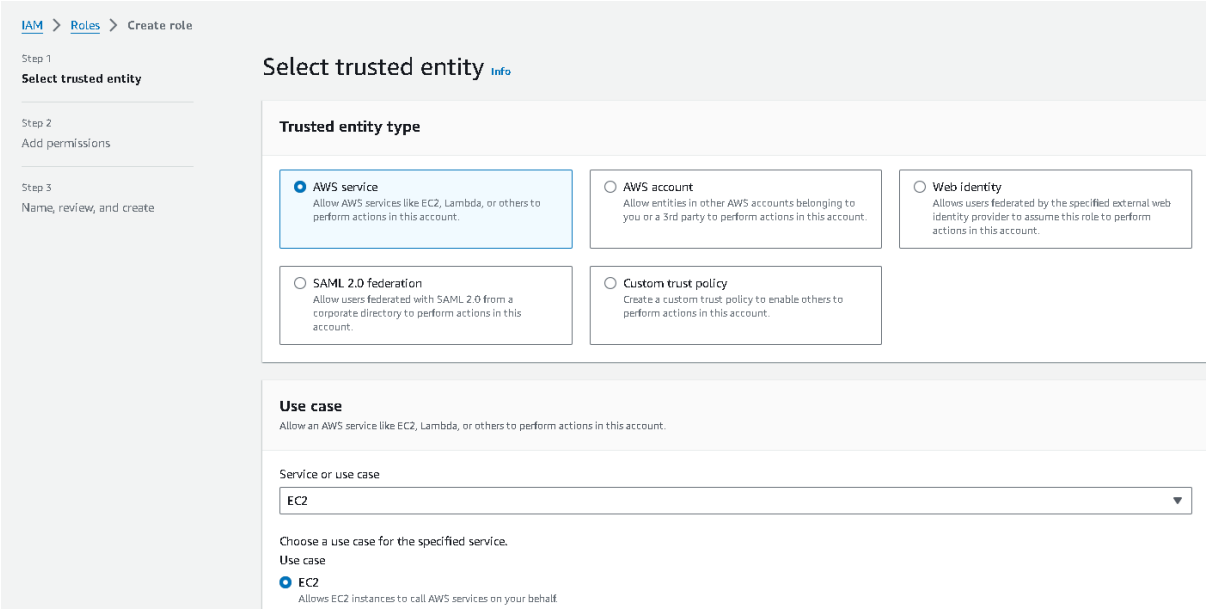
- The tasks are running successfully:



STEP NO: 7

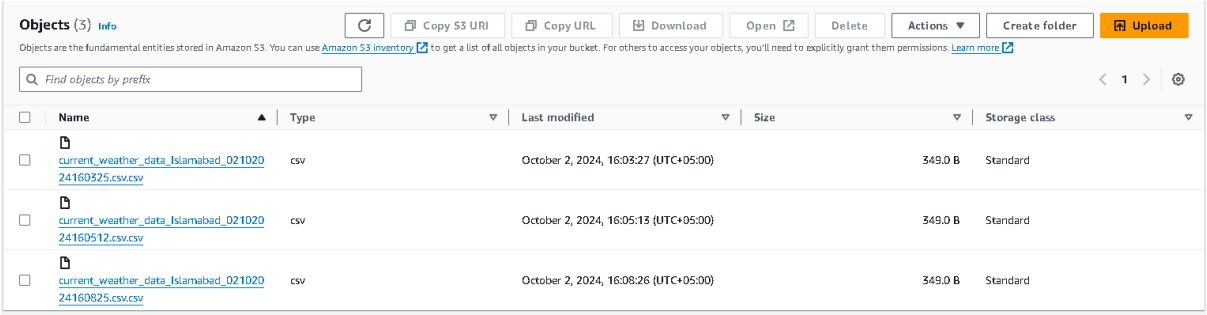
- Now creating our aws S3 bucket and from EC2 instance give permission
  To the S3

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



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!



THE END