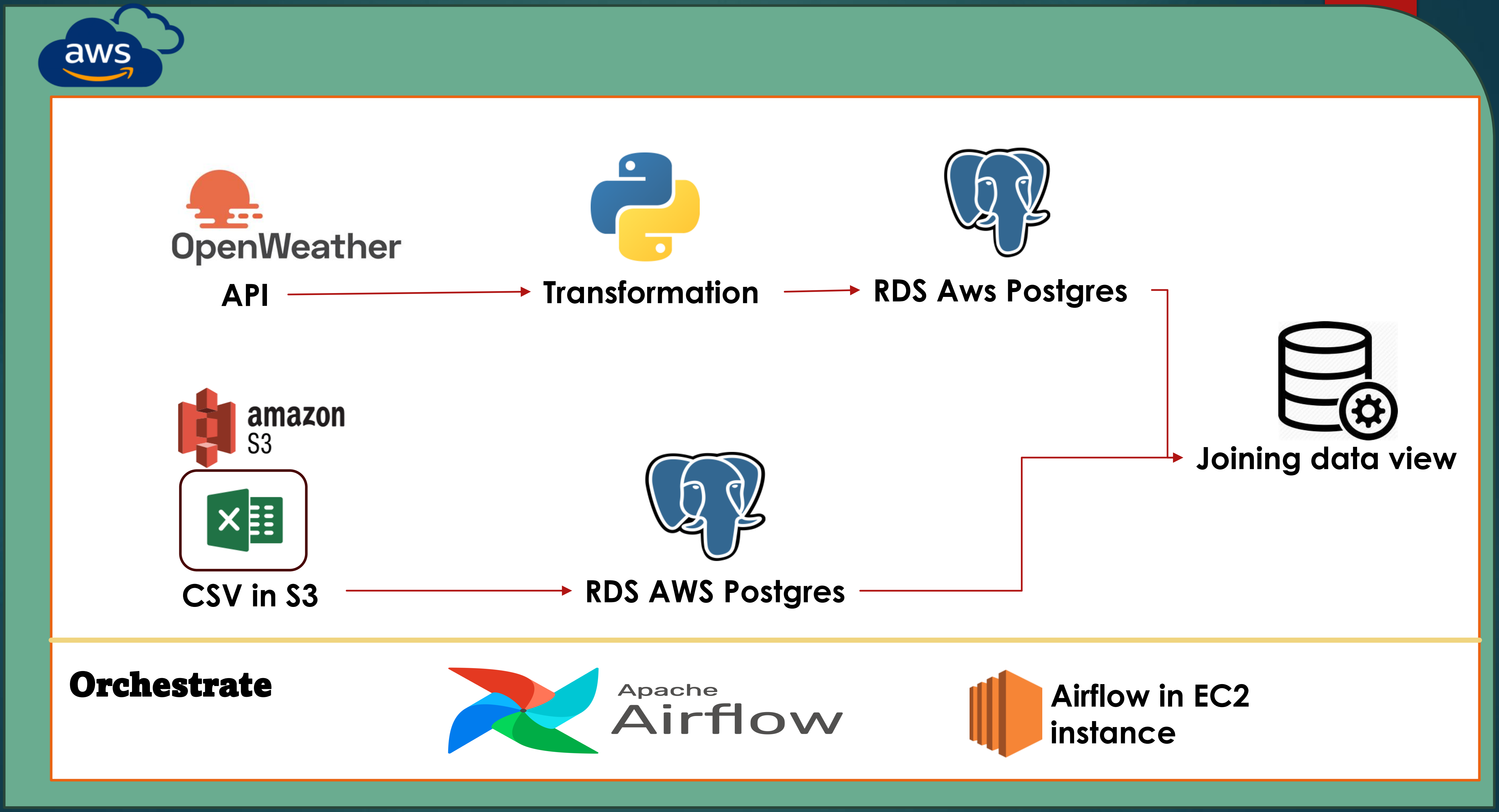
WEATHER AIRFLOW PROJECT

BY :

TUQA HUSSIEN
ABDALLAH AMR

Apache Airflow

Weather Project Architecture

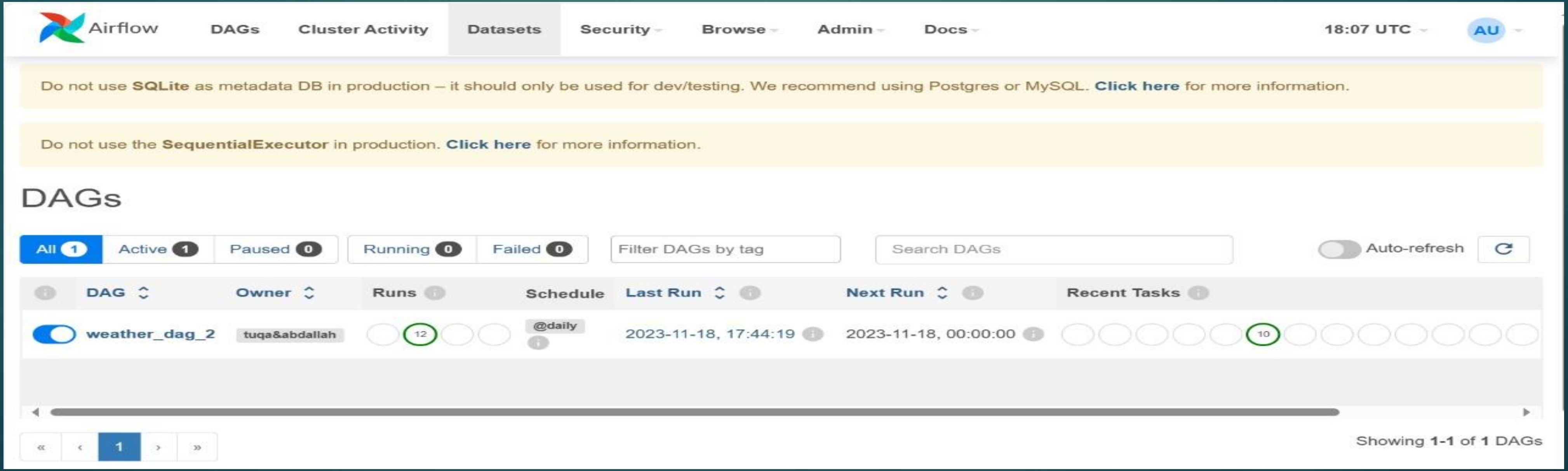


WEATHER AIRFLOW PROJECT

This project automates the retrieval, transformation, and storage of weather data from an API and csv file in s3 into a PostgreSQL database using Apache Airflow Deployed in ec2 instance .

Overview

This contains code to set up an Airflow Directed Acyclic Graph (DAG) named `weather_dag.py`. This DAG orchestrates tasks to:



Do not use **SQLite** as metadata DB in production – it should only be used for dev/testing. We recommend using Postgres or MySQL. **Click here** for more information.

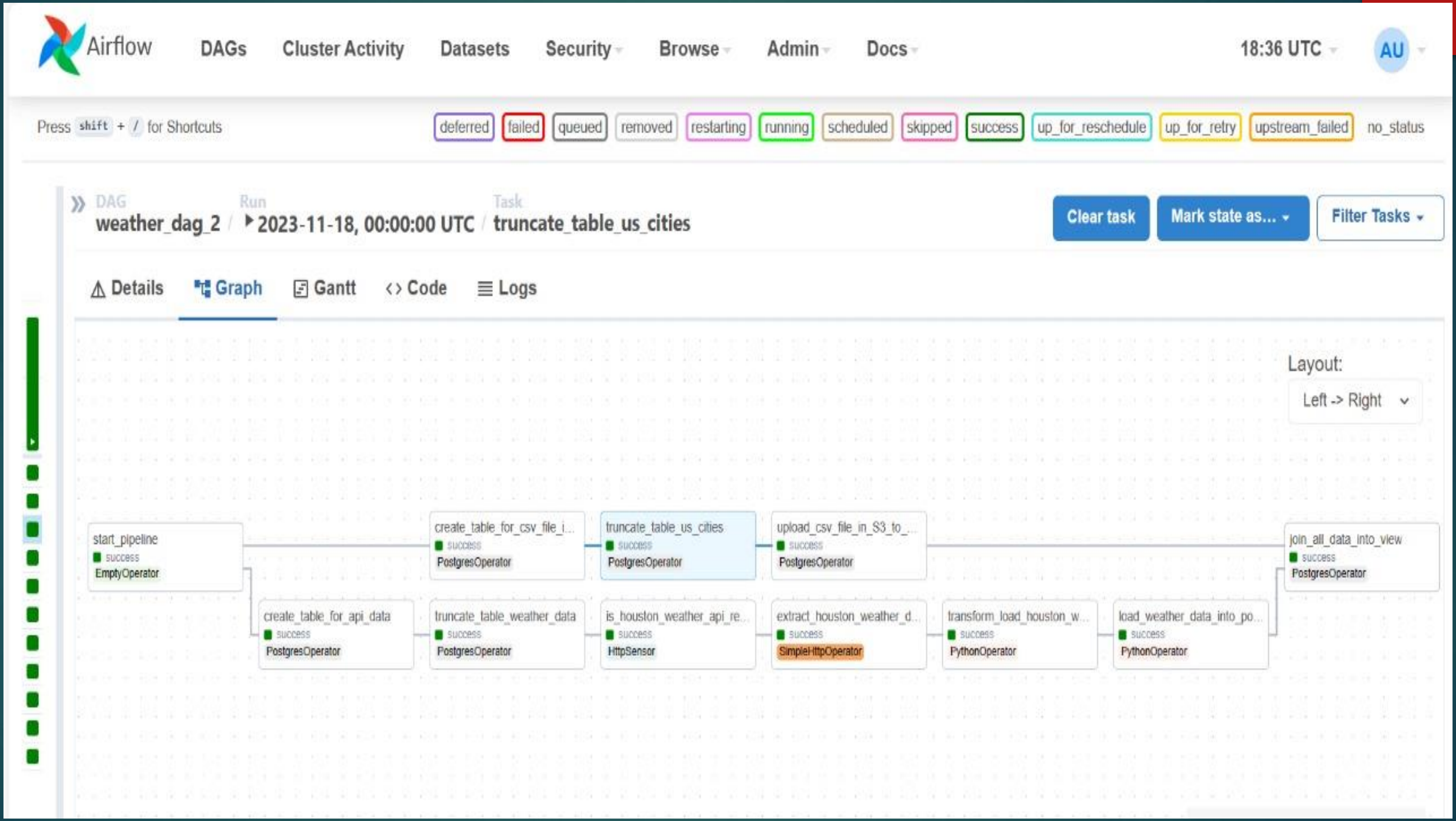
Do not use the **SequentialExecutor** in production. **Click here** for more information.

DAGs

All 1 | Active 1 | Paused 0 | Running 0 | Failed 0 | Filter DAGs by tag | Search DAGs | Auto-refresh

DAG	Owner	Runs	Schedule	Last Run	Next Run	Recent Tasks
weather_dag_2	tuqa&abdallah	12	@daily	2023-11-18, 17:44:19	2023-11-18, 00:00:00	10

Showing 1-1 of 1 DAGs

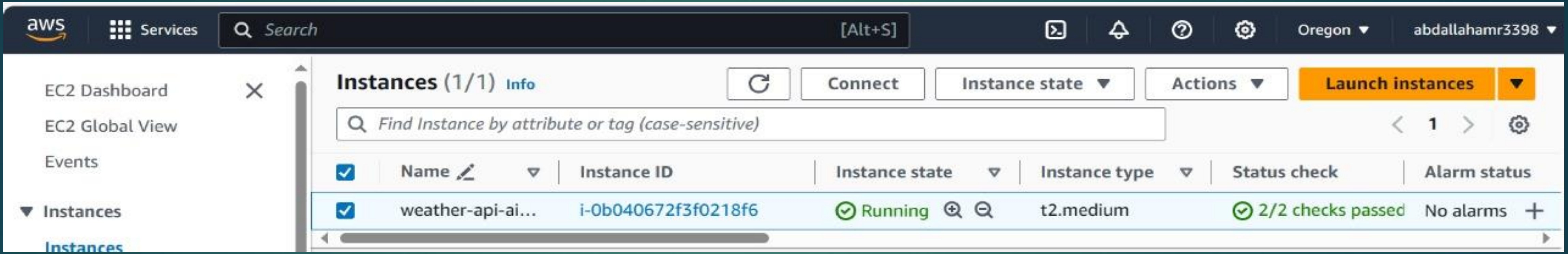


DAG weather_dag_2 / Run 2023-11-18, 00:00:00 UTC / Task truncate_table_us_cities

Layout: Left -> Right

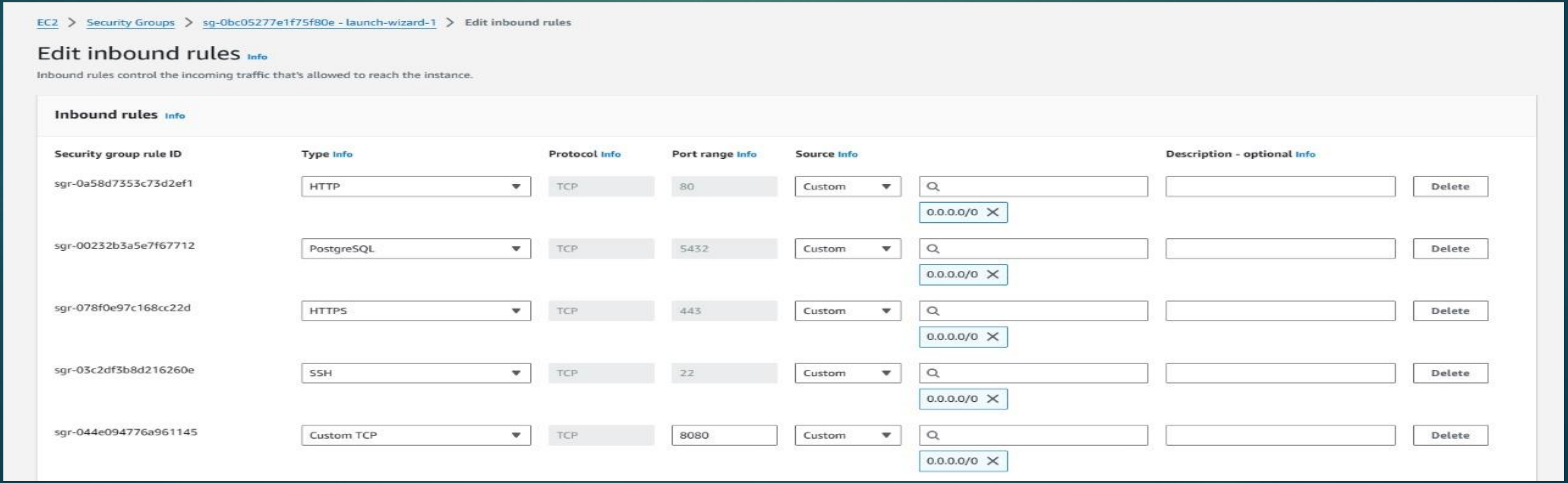
project steps :

1-Create EC2 instance with it's security Group In AWS Cloud and install (Airflow)



Instances (1/1) Info

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status
✓	weather-api-ai...	i-0b040672f3f0218f6	Running	t2.medium	2/2 checks passed	No alarms



EC2 > Security Groups > sg-0bc05277e1f75f80e - launch-wizard-1 > Edit inbound rules

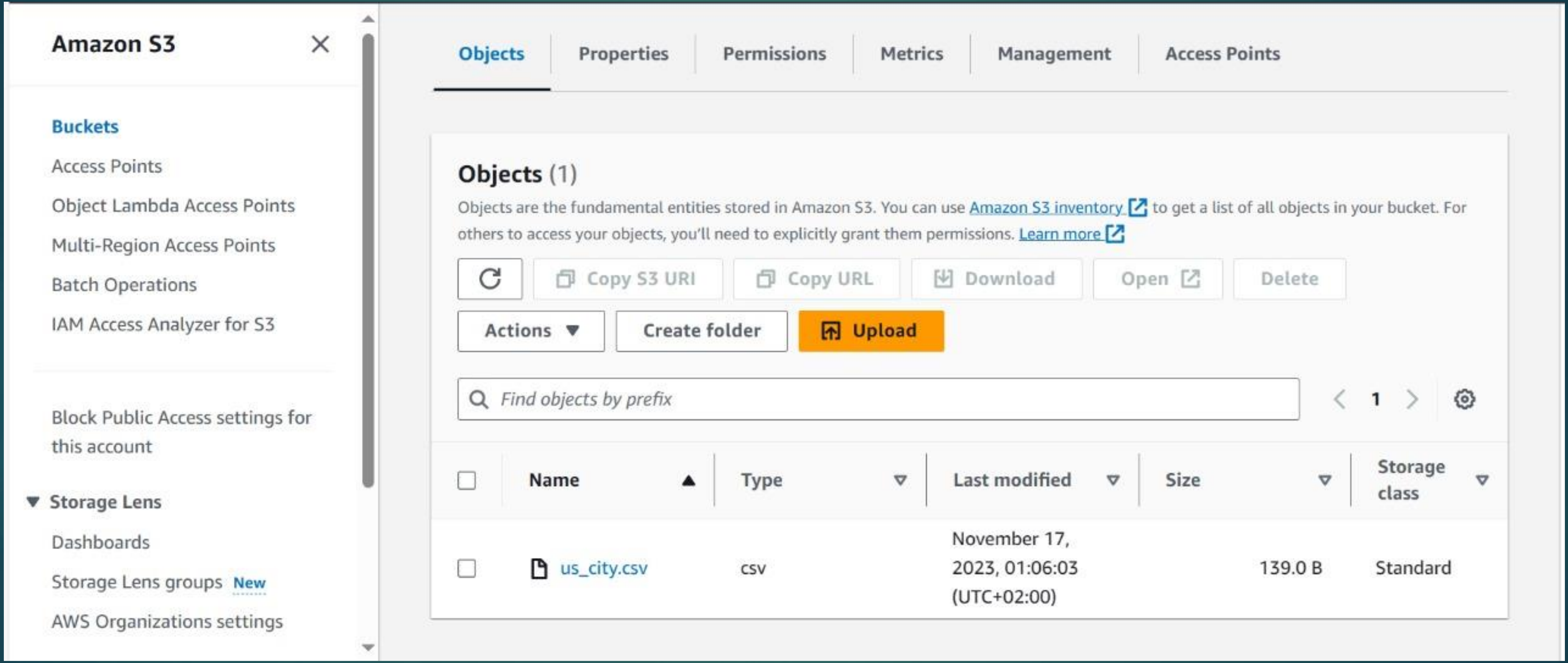
Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
sgr-0a58d7353c73d2ef1	HTTP	TCP	80	Custom	0.0.0.0/0		Delete
sgr-00232b3a5e7f67712	PostgreSQL	TCP	5432	Custom	0.0.0.0/0		Delete
sgr-078f0e97c168cc22d	HTTPS	TCP	443	Custom	0.0.0.0/0		Delete
sgr-03c2df3b8d216260e	SSH	TCP	22	Custom	0.0.0.0/0		Delete
sgr-044e094776a961145	Custom TCP	TCP	8080	Custom	0.0.0.0/0		Delete

2- Create S3 upload file CSV

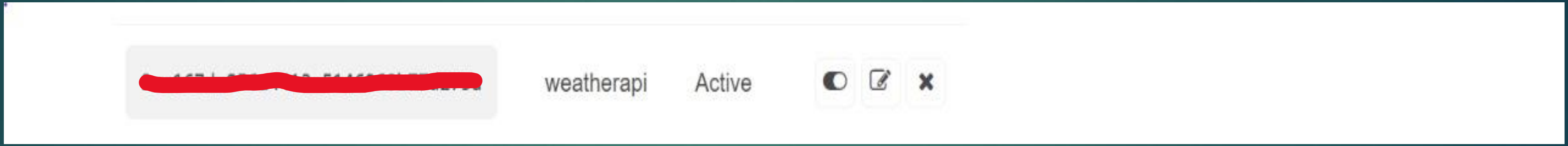


Objects (1)

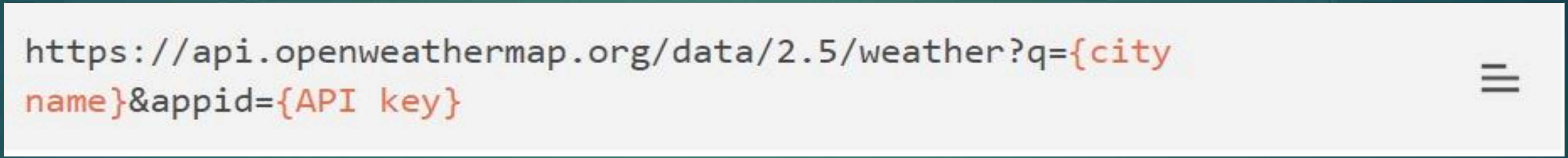
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
us_city.csv	csv	November 17, 2023, 01:06:03 (UTC+02:00)	139.0 B	Standard

3- Generate from open weather API key

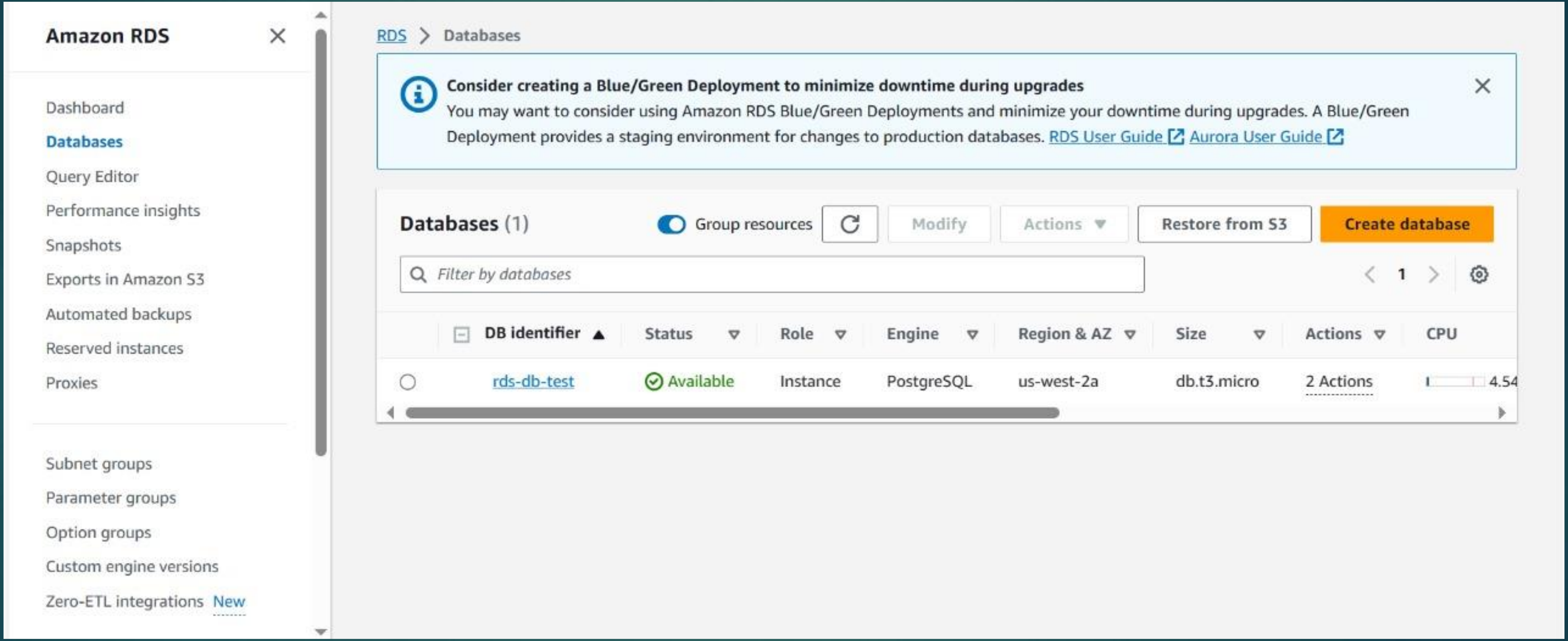


[redacted] weatherapi Active



```
https://api.openweathermap.org/data/2.5/weather?q={city name}&appid={API key}
```

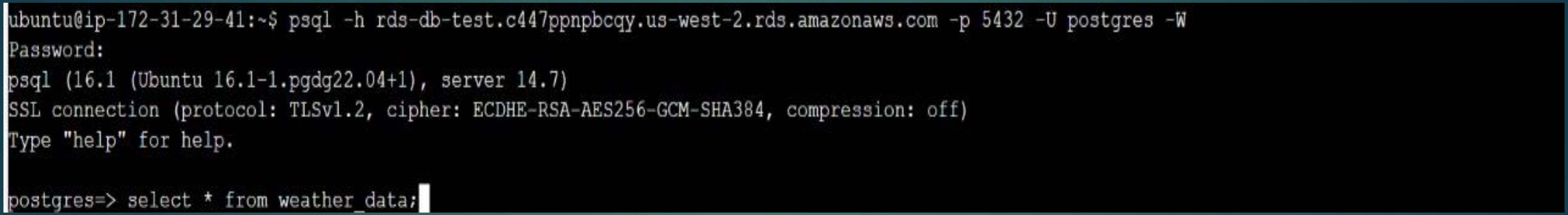
4-Create RDS Postgres Database



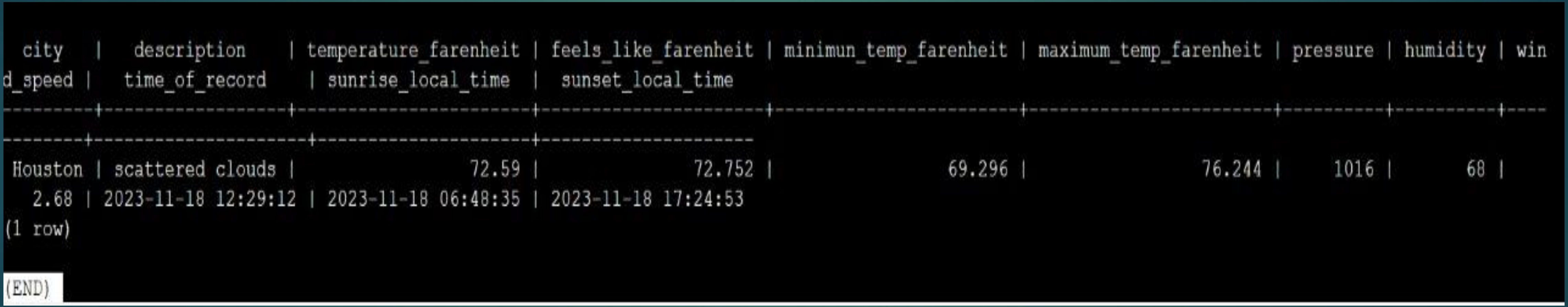
Consider creating a Blue/Green Deployment to minimize downtime during upgrades
You may want to consider using Amazon RDS Blue/Green Deployments and minimize your downtime during upgrades. A Blue/Green Deployment provides a staging environment for changes to production databases. RDS User Guide Aurora User Guide

DB identifier	Status	Role	Engine	Region & AZ	Size	Actions	CPU
rds-db-test	Available	Instance	PostgreSQL	us-west-2a	db.t3.micro	2 Actions	4.54

5-Load Api Data into postgres Table

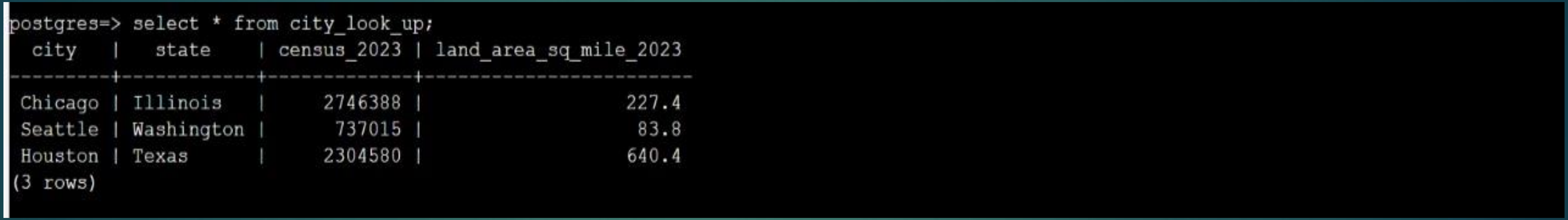


```
ubuntu@ip-172-31-29-41:~$ psql -h rds-db-test.c447ppnpbcqy.us-west-2.rds.amazonaws.com -p 5432 -U postgres -W
Password:
psql (16.1 (Ubuntu 16.1-1.pgdg22.04+1), server 14.7)
SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA-AES256-GCM-SHA384, compression: off)
Type "help" for help.

postgres=> select * from weather_data;
```



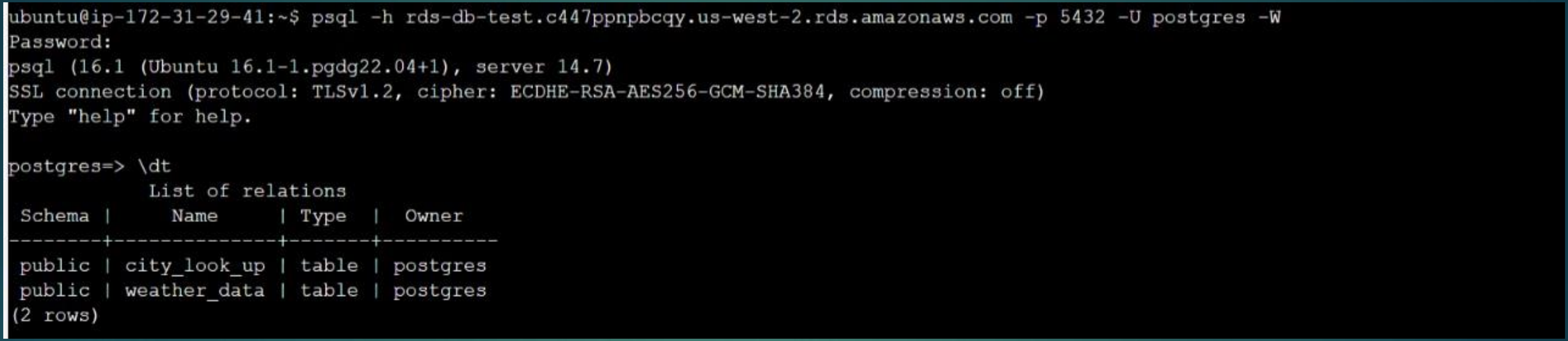
```
  city    |    description     | temperature_farenheit | feels_like_farenheit | minimun_temp_farenheit | maximum_temp_farenheit | pressure | humidity | win
d_speed |    time_of_record     | sunrise_local_time    |  sunset_local_time
---------+------------------+----------------------+---------------------+----------------------+----------------------+----------+----------+----
--------+---------------------+---------------------+---------------------
 Houston | scattered clouds |                72.59 |              72.752 |                69.296 |                76.244 |     1016 |       68 |
   2.68 | 2023-11-18 12:29:12 | 2023-11-18 06:48:35 | 2023-11-18 17:24:53
(1 row)

(END)
```

6-Load csv file into postgres Table

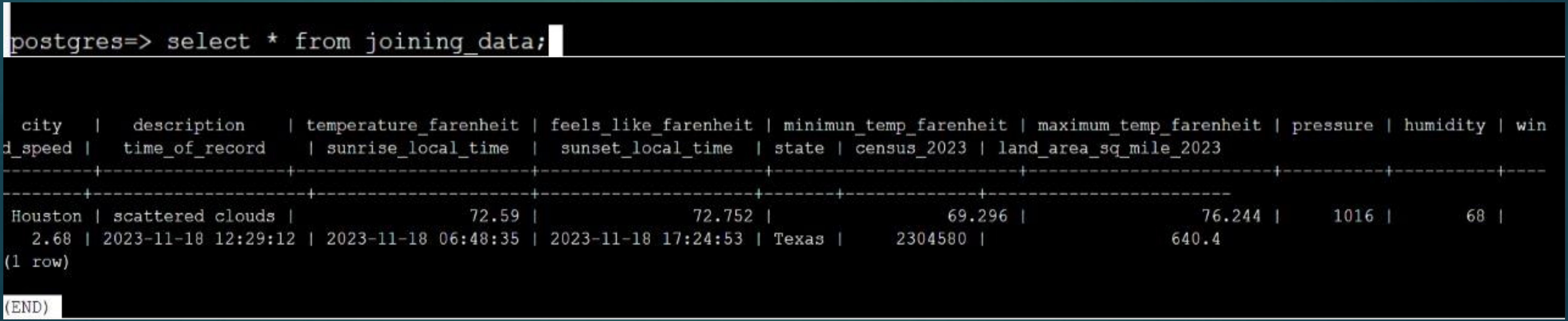


```
postgres=> select * from city_look_up;
  city   |   state    | census_2023 | land_area_sq_mile_2023
---------+------------+-------------+------------------------
 Chicago | Illinois   |     2746388 |                  227.4
 Seattle | Washington |      737015 |                   83.8
 Houston | Texas      |     2304580 |                  640.4
(3 rows)
```

7-select from joining_data view



```
ubuntu@ip-172-31-29-41:~$ psql -h rds-db-test.c447ppnpbcqy.us-west-2.rds.amazonaws.com -p 5432 -U postgres -W
Password:
psql (16.1 (Ubuntu 16.1-1.pgdg22.04+1), server 14.7)
SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA-AES256-GCM-SHA384, compression: off)
Type "help" for help.

postgres=> \dt
           List of relations
 Schema |     Name     | Type  |  Owner
--------+--------------+-------+----------
 public | city_look_up | table | postgres
 public | weather_data | table | postgres
(2 rows)
```



```
postgres=> select * from joining_data;


  city   |   description    | temperature_farenheit | feels_like_farenheit | minimun_temp_farenheit | maximum_temp_farenheit | pressure | humidity | win
d_speed |   time_of_record    | sunrise_local_time  |  sunset_local_time  | state | census_2023 | land_area_sq_mile_2023
---------+------------------+-----------------------+----------------------+------------------------+------------------------+----------+----------+----
--------+---------------------+---------------------+---------------------+-------+-------------+------------------------
 Houston | scattered clouds |                 72.59 |               72.752 |                 69.296 |                 76.244 |     1016 |       68 |
   2.68 | 2023-11-18 12:29:12 | 2023-11-18 06:48:35 | 2023-11-18 17:24:53 | Texas |     2304580 |                  640.4
(1 row)

(END)
```

Thank You