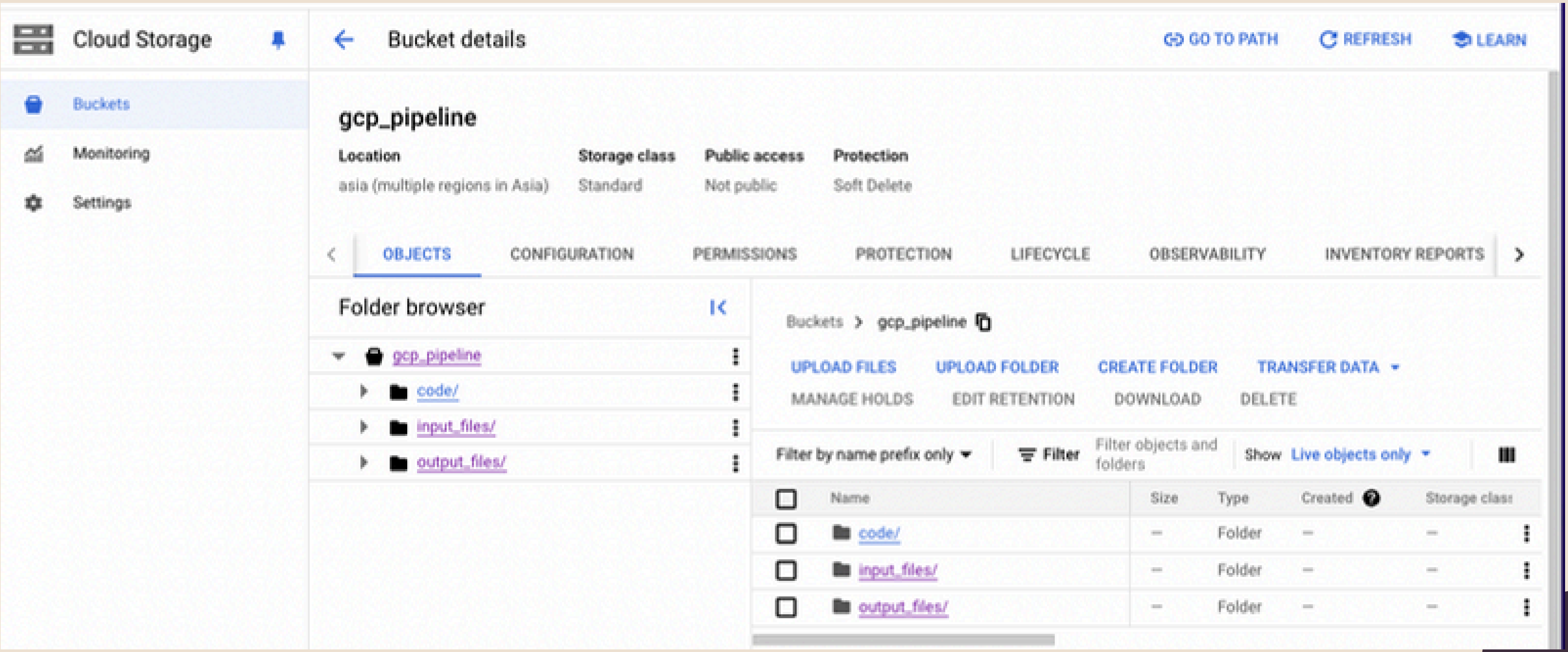
ETL GCP-BIGQUERY

Then Tsze Yen

STEP 1

-Create GCP Cloud Storage Bucket: gcp_pipeline

-Load data into GCP Cloud Storage: input_files/



gcp_pipeline

Location	Storage class	Public access	Protection
asia (multiple regions in Asia)	Standard	Not public	Soft Delete

gcp_pipeline
code/
input_files/
output_files/

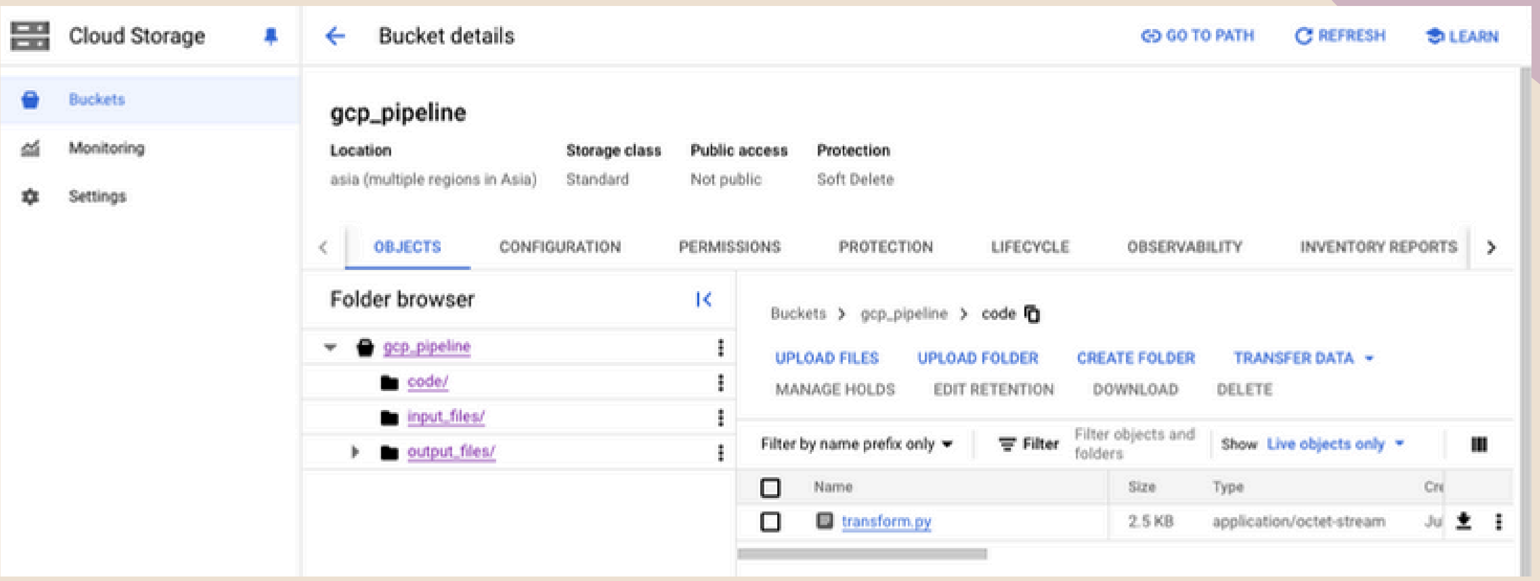
Buckets > gcp_pipeline

□	Name	Size	Type	Created	Storage class
□	code/	–	Folder	–	–
□	input_files/	–	Folder	–	–
□	output_files/	–	Folder	–	–

STEP 2

-Load the transform.py to code file (created your own)

1



gcp_pipeline

Location: asia (multiple regions in Asia)
Storage class: Standard
Public access: Not public
Protection: Soft Delete

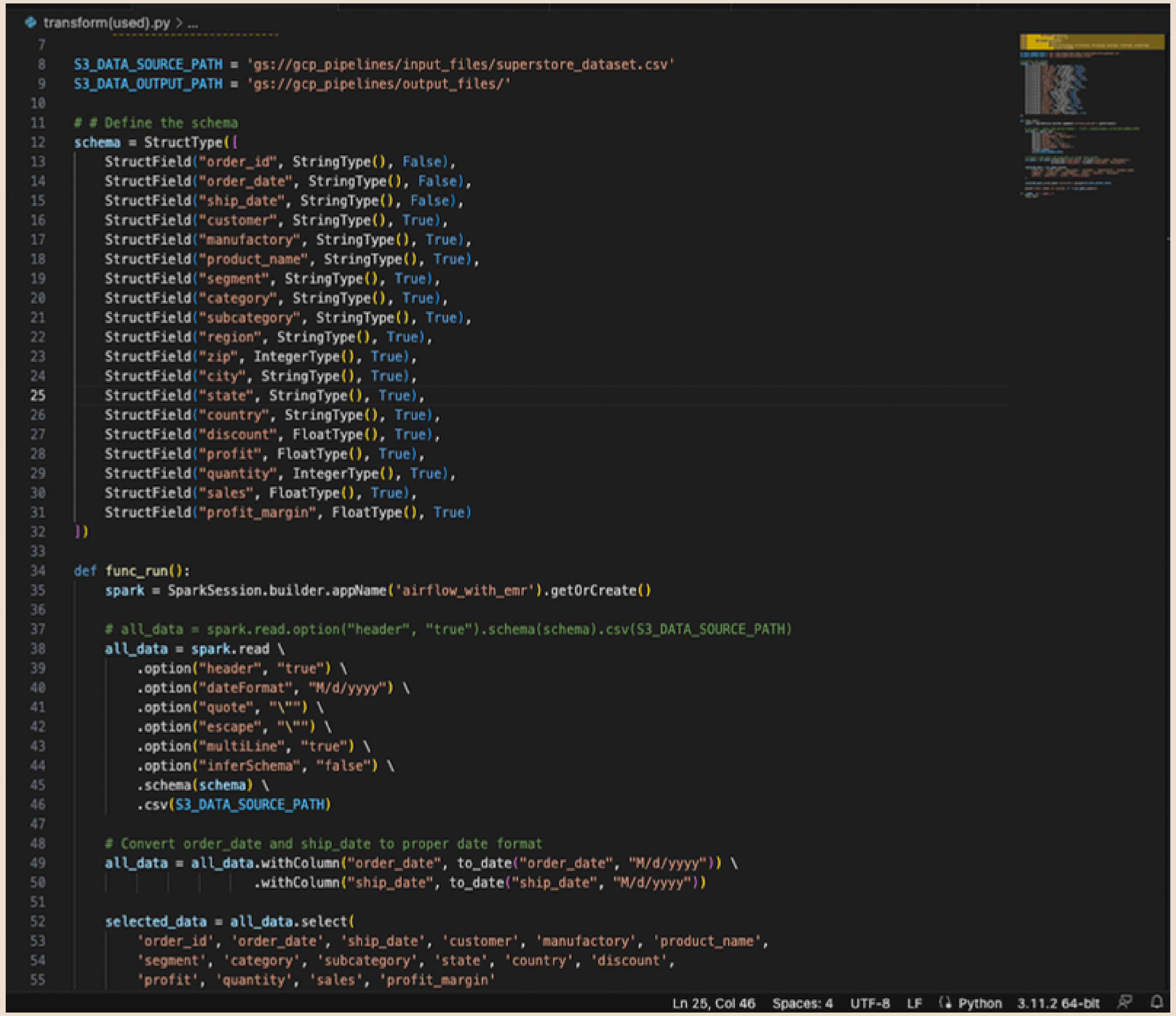
gcp_pipeline
- code/
- input_files/
- output_files/

Buckets > gcp_pipeline > code

Show Live objects only

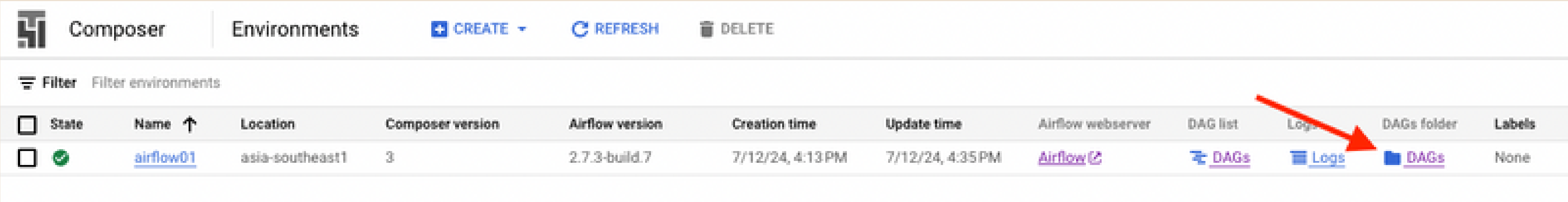
☐	Name	Size	Type	Cre
☐	transform.py	2.5 KB	application/octet-stream	Ju

2


transform(used).py > ...

```python

S3_DATA_SOURCE_PATH = 'gs://gcp_pipelines/input_files/superstore_dataset.csv'
S3_DATA_OUTPUT_PATH = 'gs://gcp_pipelines/output_files/'

# # Define the schema
schema = StructType([
    StructField("order_id", StringType(), False),
    StructField("order_date", StringType(), False),
    StructField("ship_date", StringType(), False),
    StructField("customer", StringType(), True),
    StructField("manufactory", StringType(), True),
    StructField("product_name", StringType(), True),
    StructField("segment", StringType(), True),
    StructField("category", StringType(), True),
    StructField("subcategory", StringType(), True),
    StructField("region", StringType(), True),
    StructField("zip", IntegerType(), True),
    StructField("city", StringType(), True),
    StructField("state", StringType(), True),
    StructField("country", StringType(), True),
    StructField("discount", FloatType(), True),
    StructField("profit", FloatType(), True),
    StructField("quantity", IntegerType(), True),
    StructField("sales", FloatType(), True),
    StructField("profit_margin", FloatType(), True)
])

def func_run():
    spark = SparkSession.builder.appName('airflow_with_emr').getOrCreate()

    # all_data = spark.read.option("header", "true").schema(schema).csv(S3_DATA_SOURCE_PATH)
    all_data = spark.read \
        .option("header", "true") \
        .option("dateFormat", "M/d/yyyy") \
        .option("quote", "\"") \
        .option("escape", "\"") \
        .option("multiLine", "true") \
        .option("inferSchema", "false") \
        .schema(schema) \
        .csv(S3_DATA_SOURCE_PATH)

    # Convert order_date and ship_date to proper date format
    all_data = all_data.withColumn("order_date", to_date("order_date", "M/d/yyyy")) \
                        .withColumn("ship_date", to_date("ship_date", "M/d/yyyy"))

    selected_data = all_data.select(
        'order_id', 'order_date', 'ship_date', 'customer', 'manufactory', 'product_name',
        'segment', 'category', 'subcategory', 'state', 'country', 'discount',
        'profit', 'quantity', 'sales', 'profit_margin'
```

Ln 25, Col 46   Spaces: 4   UTF-8   LF   Python   3.11.2 64-bit

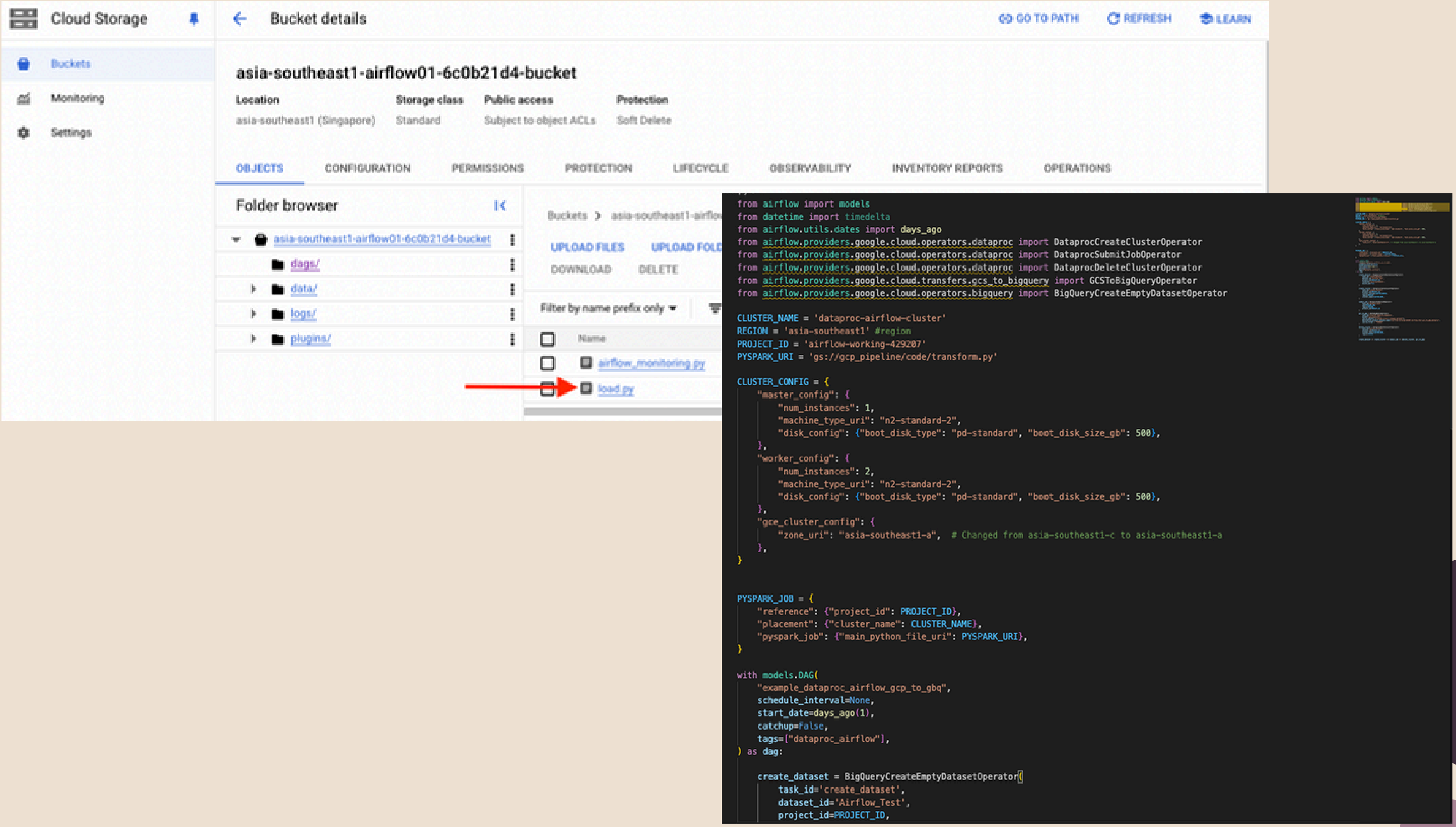


1

Composer | Environments | CREATE | REFRESH | DELETE

Filter | Filter environments

☐	State	Name ↑	Location	Composer version	Airflow version	Creation time	Update time	Airflow webserver	DAG list	Logs	DAGs folder	Labels
☐	✓	airflow01	asia-southeast1	3	2.7.3-build.7	7/12/24, 4:13 PM	7/12/24, 4:35 PM	Airflow	DAGs	Logs	DAGs	None



2

Cloud Storage

- Buckets
- Monitoring
- Settings

Bucket details | GO TO PATH | REFRESH | LEARN

asia-southeast1-airflow01-6c0b21d4-bucket

Location	Storage class	Public access	Protection
asia-southeast1 (Singapore)	Standard	Subject to object ACLs	Soft Delete

OBJECTS | CONFIGURATION | PERMISSIONS | PROTECTION | LIFECYCLE | OBSERVABILITY | INVENTORY REPORTS | OPERATIONS

Folder browser

- asia-southeast1-airflow01-6c0b21d4-bucket
  - dags/
  - data/
  - logs/
  - plugins/

Buckets > asia-southeast1-airflo

UPLOAD FILES | UPLOAD FOLD | DOWNLOAD | DELETE

Filter by name prefix only

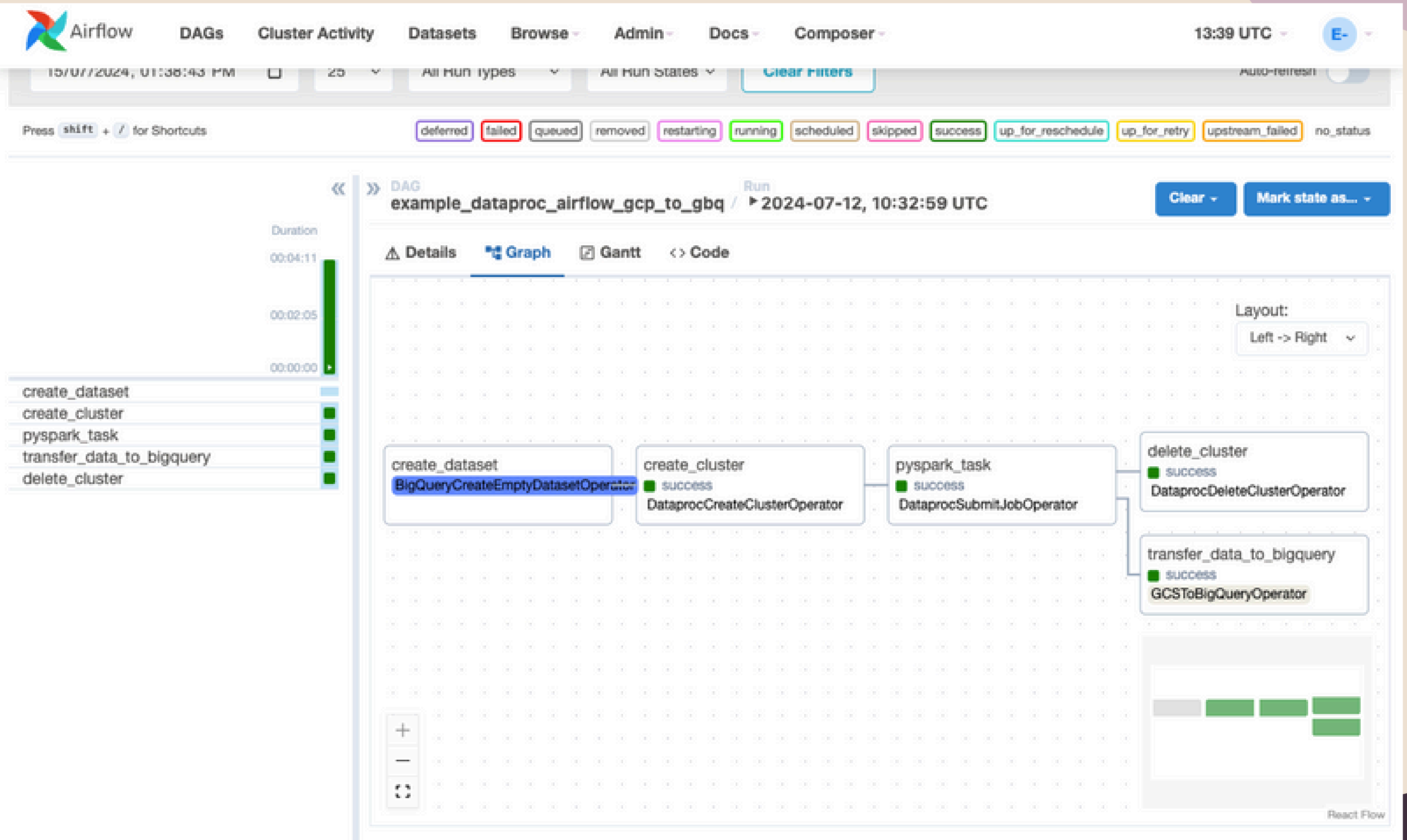
☐	Name
☐	airflow_monitoring.py
☐	load.py

```python
from airflow import models
from datetime import timedelta
from airflow.utils.dates import days_ago
from airflow.providers.google.cloud.operators.dataproc import DataprocCreateClusterOperator
from airflow.providers.google.cloud.operators.dataproc import DataprocSubmitJobOperator
from airflow.providers.google.cloud.operators.dataproc import DataprocDeleteClusterOperator
from airflow.providers.google.cloud.transfers.gcs_to_bigquery import GCSToBigQueryOperator
from airflow.providers.google.cloud.operators.bigquery import BigQueryCreateEmptyDatasetOperator

CLUSTER_NAME = 'dataproc-airflow-cluster'
REGION = 'asia-southeast1' #region
PROJECT_ID = 'airflow-working-429207'
PYSPARK_URI = 'gs://gcp_pipeline/code/transform.py'

CLUSTER_CONFIG = {
    "master_config": {
        "num_instances": 1,
        "machine_type_uri": "n2-standard-2",
        "disk_config": {"boot_disk_type": "pd-standard", "boot_disk_size_gb": 500},
    },
    "worker_config": {
        "num_instances": 2,
        "machine_type_uri": "n2-standard-2",
        "disk_config": {"boot_disk_type": "pd-standard", "boot_disk_size_gb": 500},
    },
    "gce_cluster_config": {
        "zone_uri": "asia-southeast1-a",  # Changed from asia-southeast1-c to asia-southeast1-a
    },
}


PYSPARK_JOB = {
    "reference": {"project_id": PROJECT_ID},
    "placement": {"cluster_name": CLUSTER_NAME},
    "pyspark_job": {"main_python_file_uri": PYSPARK_URI},
}

with models.DAG(
    "example_dataproc_airflow_gcp_to_gbq",
    schedule_interval=None,
    start_date=days_ago(1),
    catchup=False,
    tags=["dataproc_airflow"],
) as dag:

    create_dataset = BigQueryCreateEmptyDatasetOperator(
        task_id='create_dataset',
        dataset_id='Airflow_Test',
        project_id=PROJECT_ID,
```

STEP 3

-Load the load.py to DAGs from GCP Composer

AIRFLOW OUTPUT



DAG
example_dataproc_airflow_gcp_to_gbq / Run
2024-07-12, 10:32:59 UTC

create_dataset
create_cluster
pyspark_task
transfer_data_to_bigquery
delete_cluster

create_dataset
BigQueryCreateEmptyDatasetOperator

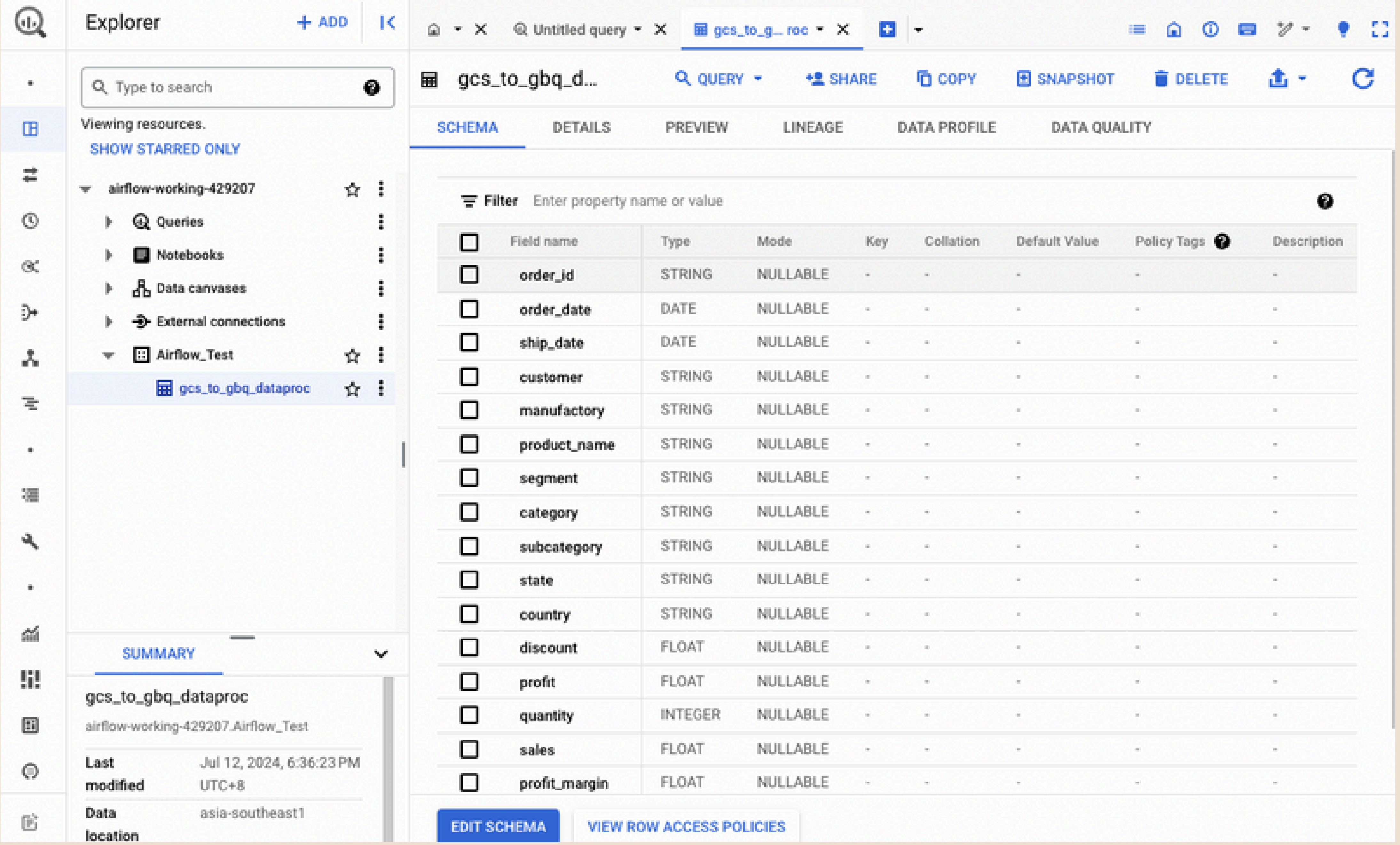
create_cluster
success
DataprocCreateClusterOperator

pyspark_task
success
DataprocSubmitJobOperator

delete_cluster
success
DataprocDeleteClusterOperator

transfer_data_to_bigquery
success
GCSToBigQueryOperator

BIGQUERY OUTPUT



□	Field name	Type	Mode	Key	Collation	Default Value	Policy Tags	Description
□	order_id	STRING	NULLABLE	-	-	-	-	-
□	order_date	DATE	NULLABLE	-	-	-	-	-
□	ship_date	DATE	NULLABLE	-	-	-	-	-
□	customer	STRING	NULLABLE	-	-	-	-	-
□	manufactory	STRING	NULLABLE	-	-	-	-	-
□	product_name	STRING	NULLABLE	-	-	-	-	-
□	segment	STRING	NULLABLE	-	-	-	-	-
□	category	STRING	NULLABLE	-	-	-	-	-
□	subcategory	STRING	NULLABLE	-	-	-	-	-
□	state	STRING	NULLABLE	-	-	-	-	-
□	country	STRING	NULLABLE	-	-	-	-	-
□	discount	FLOAT	NULLABLE	-	-	-	-	-
□	profit	FLOAT	NULLABLE	-	-	-	-	-
□	quantity	INTEGER	NULLABLE	-	-	-	-	-
□	sales	FLOAT	NULLABLE	-	-	-	-	-
□	profit_margin	FLOAT	NULLABLE	-	-	-	-	-

gcs_to_gbq_dataproc

airflow-working-429207.Airflow_Test

Last modified	Jul 12, 2024, 6:36:23 PM UTC+8
Data location	asia-southeast1

THANK YOU