📘 Airflow Assignment 2 – Load JSON Data from GCS to Hive on Dataproc

---

📝 Objective

This assignment demonstrates the use of **Apache Airflow** to automate a data pipeline that:

- Fetches a daily JSON file (Employee.json) from a **Google Cloud Storage (GCS)** bucket.
- Triggers a **PySpark job on Dataproc** that:
  - Creates a Hive database and table (if not already present).
  - Reads the data.
  - Appends it into a Hive-managed table in **Parquet** format.

⚙️ Components Involved

Component	Description
Airflow	Orchestrates the DAG tasks
GCSToLocalFilesystemOperator	Downloads JSON file from GCS to local /tmp/ (for completeness)
DataprocSubmitJobOperator	Submits the Spark job that writes data to Hive
Google Cloud Storage	Stores the input JSON file and PySpark script
Google Dataproc	Executes the Spark job on a managed cluster
Hive (on Dataproc)	Stores the final table (EMP_DB.employee) in Parquet format

---

🗂️ File Structure

bash
CopyEdit

```
.
├── dags/
│   ├── Load_json_to_hive.py          # DAG file
│   ├── spark_job/
│   │   └── spark_job.py              # PySpark job script (writee function)
│   ├── data/
│   │   └── Employee.json             # Input JSON file (stored in GCS)
│   └── DAG_flow.png                  # Visual DAG diagram
```

⚡ DAG Task Flow

1. **Download_File**
   - Uses GCSToLocalFilesystemOperator
   - Downloads Employee.json from:

bash
CopyEdit
gs://spark_ex_airflow/assignment2/data/Employee.json

- Saves it as:

  bash
  CopyEdit
  /tmp/employee.json

2. **Upload_to_Hive**
   - Uses DataprocSubmitJobOperator
   - Submits a PySpark job to:
     - Create database EMP_DB
     - Create table employee (if not exists)
     - Load and append data from the GCS path
     - Save it in Hive in Parquet format

🔧 Configuration Details

DataprocSubmitJobOperator

```
job={
   "placement": {"cluster_name": "dataproc-hive"},
   "pyspark_job": {
      "main_python_file_uri": "gs://spark_ex_airflow/assignment2/spark_job/spark_job.py"
   },
}
```

Spark Code Summary

- Reads data from GCS:

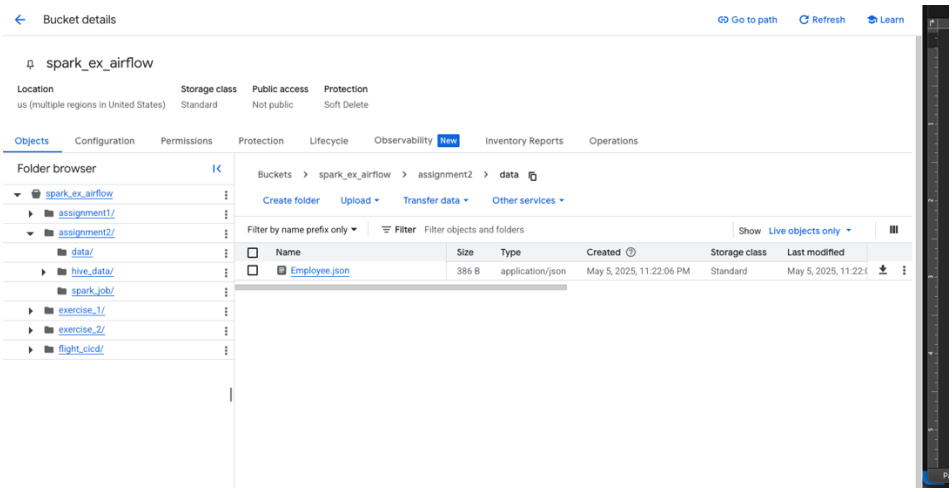
  input_path = "gs://spark_ex_airflow/assignment2/data/"

- Writes to Hive:

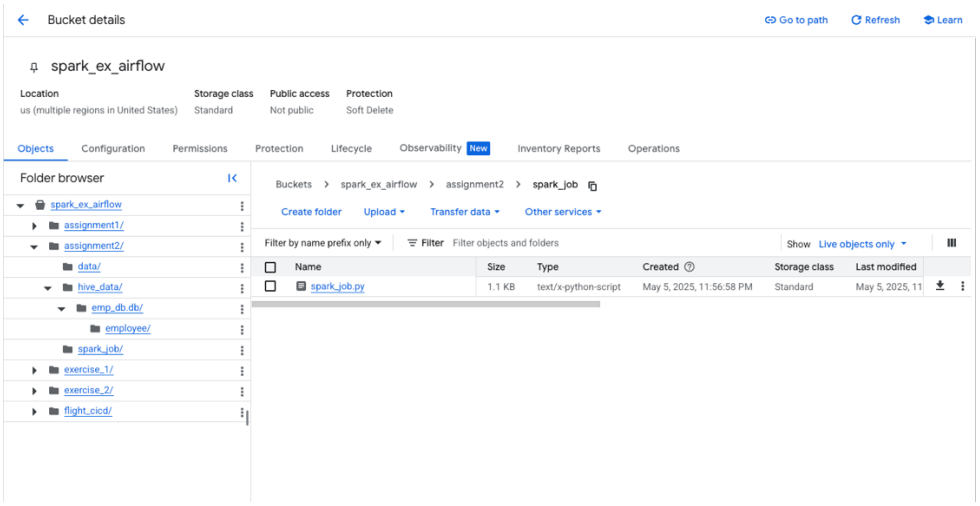
  df.write.mode("append").format("hive").saveAsTable("employee")

📋 How to Execute

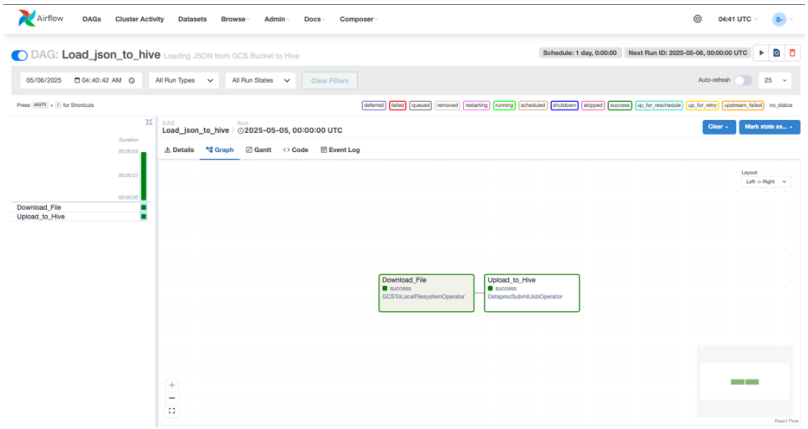
1. **Upload Employee.json** to: gs://spark_ex_airflow/assignment2/data/Employee.json



2. **Upload spark_job.py** to: gs://spark_ex_airflow/assignment2/spark_job/spark_job.py



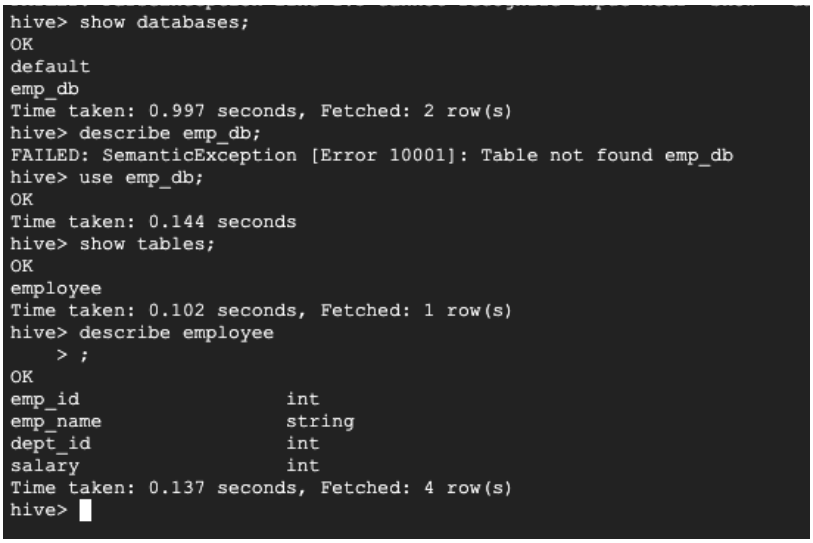
3. **Deploy DAG** to Composer or local Airflow in dags/Load_json_to_hive.py.
4. **Trigger the DAG** from the Airflow UI



✅ Expected Output

The following Hive table will be created and appended to: EMP_DB.employee

Stored as a **Parquet**-formatted managed Hive table.



⚠️ Issues Faced and Resolved

- 404 Error: GCS path was incorrect (assignment2/Employee.json instead of assignment2/data/Employee.json) → ✅ Resolved by correcting the object_name.
- Spark writing to wrong database → ✅ Fixed by explicitly using EMP_DB.employee in saveAsTable.
- GCS file not found → ✅ Reuploaded file and verified path using gsutil.