Clase 8

Consigna: Por cada ejercicio, escribir el código y agregar una captura de pantalla del resultado obtenido.

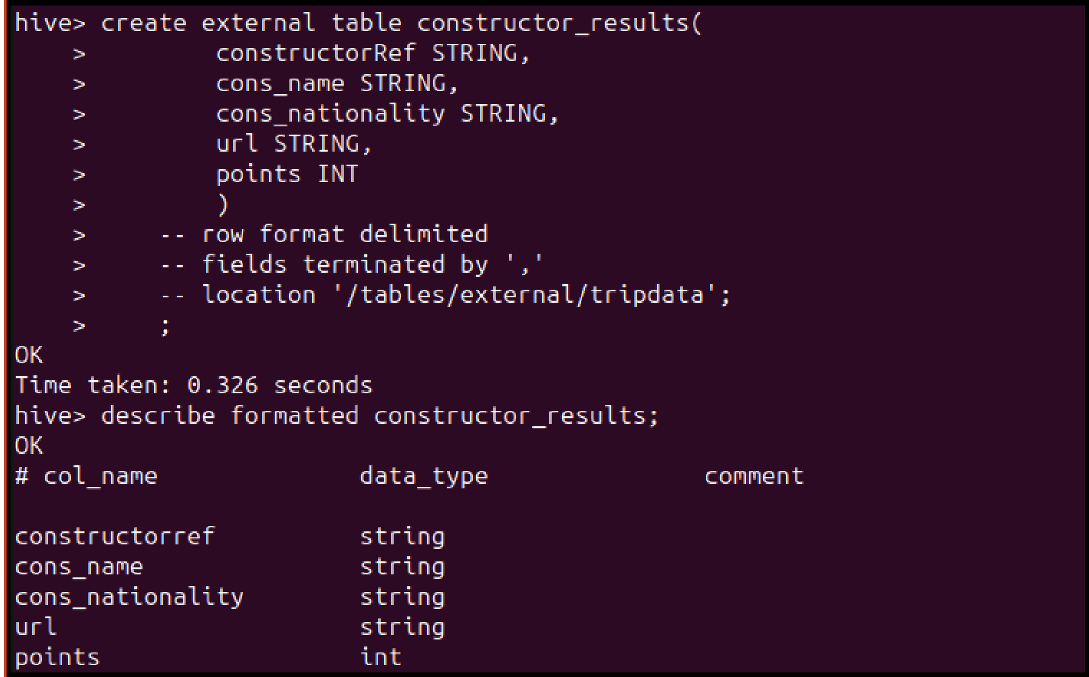
Diccionario de datos:
https://www.kaggle.com/datasets/rohanrao/formula-1-world-championship-1950-2020?select=results.csv

1. Crear la siguientes tablas externas en la base de datos f1 en hive:
   a. driver_results (driver_forename, driver_surname, driver_nationality, points) b. constructor_results (constructorRef, cons_name, cons_nationality, url, points)

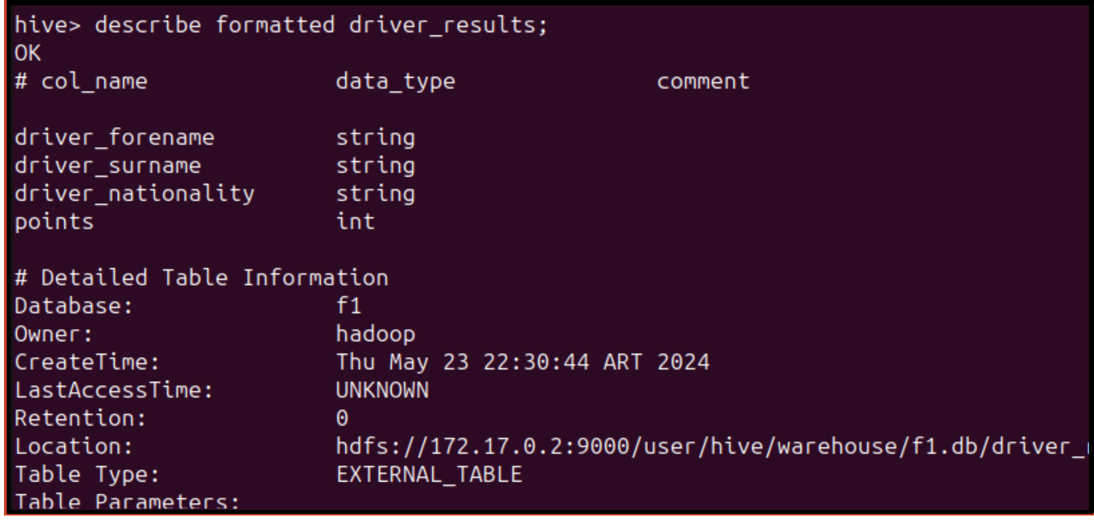
```
create database f1;

create external table driver_results(
        driver_forename STRING,
        driver_surname STRING,
        driver_nationality STRING,
        points INT
        )
    -- row format delimited
    -- fields terminated by ','
    -- location '/tables/external/tripdata';


create external table constructor_results(
        constructorRef STRING,
        cons_name STRING,
        cons_nationality STRING,
        url STRING,
        points INT
        )
    -- row format delimited
    -- fields terminated by ','
    -- location '/tables/external/tripdata';
    ;
```

2. En Hive, mostrar el esquema de driver_results y constructor_results

```
describe formatted driver_results;
describe formatted constructor_results;
```



```
hive> create external table constructor_results(
    >         constructorRef STRING,
    >         cons_name STRING,
    >         cons_nationality STRING,
    >         url STRING,
    >         points INT
    >         )
    >     -- row format delimited
    >     -- fields terminated by ','
    >     -- location '/tables/external/tripdata';
    >     ;
OK
Time taken: 0.326 seconds
hive> describe formatted constructor_results;
OK
# col_name              data_type               comment

constructorref          string
cons_name               string
cons_nationality        string
url                     string
points                  int
```



```
hive> describe formatted driver_results;
OK
# col_name              data_type               comment

driver_forename         string
driver_surname          string
driver_nationality      string
points                  int

# Detailed Table Information
Database:               f1
Owner:                  hadoop
CreateTime:             Thu May 23 22:30:44 ART 2024
LastAccessTime:         UNKNOWN
Retention:              0
Location:               hdfs://172.17.0.2:9000/user/hive/warehouse/f1.db/driver_
Table Type:             EXTERNAL_TABLE
Table Parameters:
```

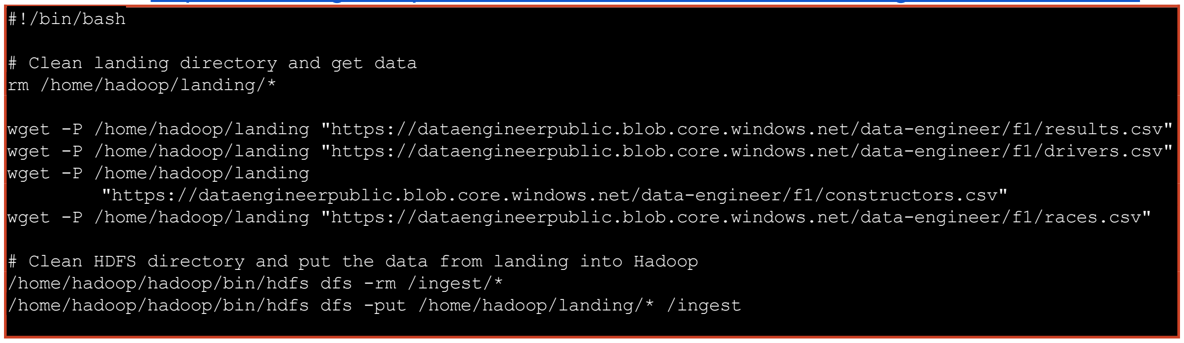
3. Crear un archivo .bash que permita descargar los archivos mencionados abajo e ingestarlos en HDFS:

    results.csv
    https://dataengineerpublic.blob.core.windows.net/data-engineer/f1/results.csv

    drivers.csv
    https://dataengineerpublic.blob.core.windows.net/data-engineer/f1/drivers.csv

    constructors.csv
    https://dataengineerpublic.blob.core.windows.net/data-engineer/f1/constructors.csv

    races.csv
    https://dataengineerpublic.blob.core.windows.net/data-engineer/f1/races.csv



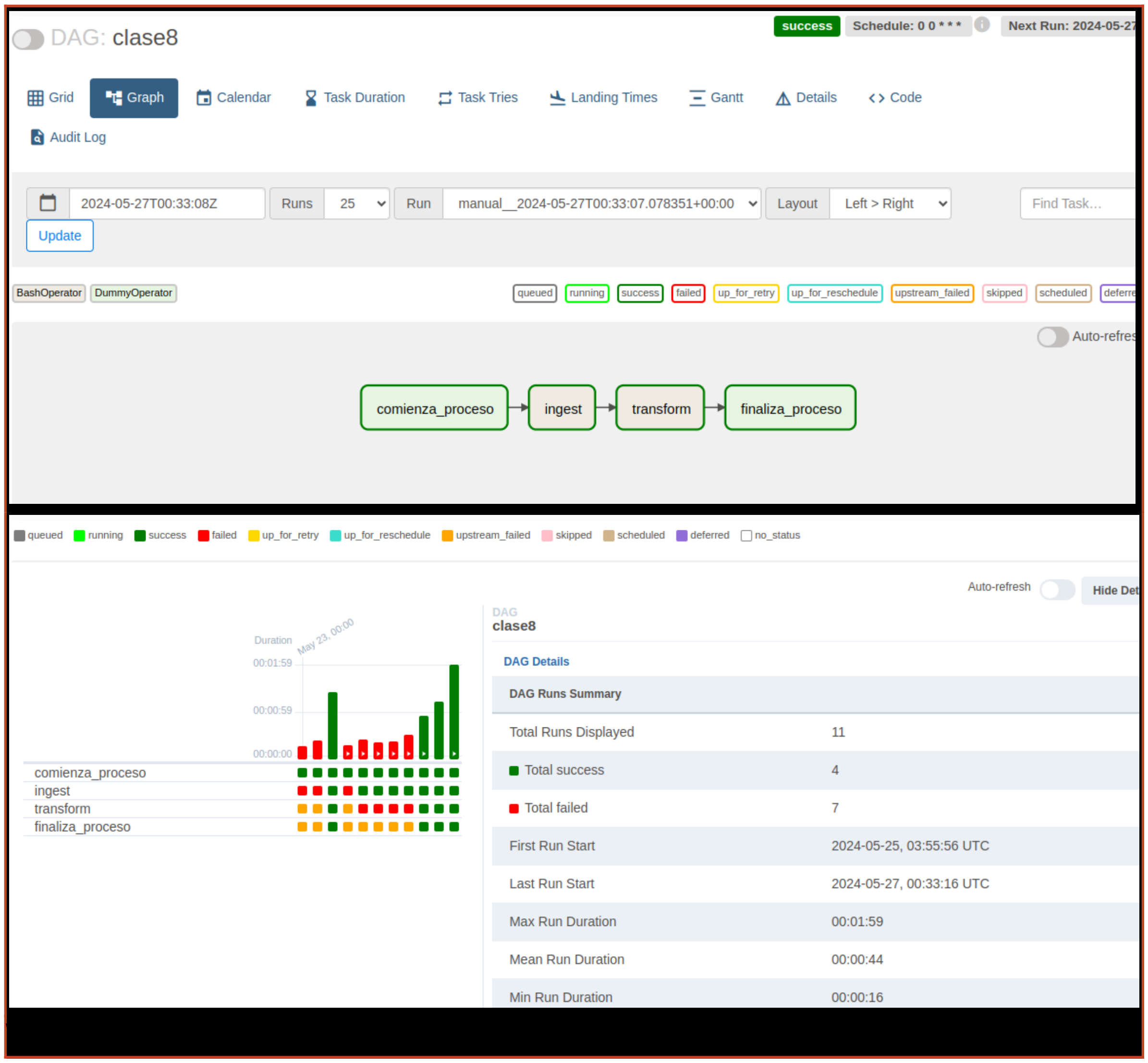
```
#!/bin/bash

# Clean landing directory and get data
rm /home/hadoop/landing/*

wget -P /home/hadoop/landing "https://dataengineerpublic.blob.core.windows.net/data-engineer/f1/results.csv"
wget -P /home/hadoop/landing "https://dataengineerpublic.blob.core.windows.net/data-engineer/f1/drivers.csv"
wget -P /home/hadoop/landing
        "https://dataengineerpublic.blob.core.windows.net/data-engineer/f1/constructors.csv"
wget -P /home/hadoop/landing "https://dataengineerpublic.blob.core.windows.net/data-engineer/f1/races.csv"

# Clean HDFS directory and put the data from landing into Hadoop
/home/hadoop/hadoop/bin/hdfs dfs -rm /ingest/*
/home/hadoop/hadoop/bin/hdfs dfs -put /home/hadoop/landing/* /ingest
```

4. Generar un archivo .py que permita, mediante Spark:
    a. insertar en la tabla driver_results los corredores con mayor cantidad de puntos en la historia.
    b. insertar en la tabla constructor_result quienes obtuvieron más puntos en el Spanish Grand Prix en el año 1991

```
from pyspark.context import SparkContext
from pyspark.sql.session import SparkSession
from pyspark.sql import HiveContext

# Configuramos...
sc = SparkContext('local')
spark = SparkSession(sc)
hc = HiveContext(sc)

# Traemos tablas
df1 = spark.read.option("header", "true").csv("hdfs://172.17.0.2:9000/ingest/results.csv")
df2 = spark.read.option("header", "true").csv("hdfs://172.17.0.2:9000/ingest/drivers.csv")
df3 = spark.read.option("header", "true").csv("hdfs://172.17.0.2:9000/ingest/constructors.csv")
df4 = spark.read.option("header", "true").csv("hdfs://172.17.0.2:9000/ingest/races.csv")

# Preparamos la primera transformacion
df1.createOrReplaceTempView("results")
df2.createOrReplaceTempView("drivers")

driver_results = spark.sql("""
    SELECT
        CAST(d.forename AS STRING) AS driver_forename,
        CAST(d.surname AS STRING) AS driver_surname,
        CAST(d.nationality AS STRING) AS driver_nationality,
        CAST(r.points AS INT) AS points
    FROM drivers d
    INNER JOIN results r ON d.driverId = r.driverId
    WHERE r.points!= 0
        """)


# Preparamos la segunda transformacion
df3.createOrReplaceTempView("constructors")
df4.createOrReplaceTempView("races")

SpanishGP = spark.sql("""
    SELECT
        CAST(c.constructorRef AS STRING) AS constructorref,
        CAST(c.name AS STRING) AS cons_name,
        CAST(c.nationality AS STRING) AS cons_nationality,
        CAST(c.url AS STRING) AS url,
        CAST(r.points AS INT) AS points
    FROM constructors c
    INNER JOIN results r ON c.constructorId = r.constructorId
    INNER JOIN  races ra ON ra.raceId = r.raceId
    WHERE ra.circuitID IN (4, 12, 26, 45, 49, 67) AND r.points != 0 AND ra.year = 1991
        """)

SpanishGP.show(3)

# Insertamos las tablas
driver_results.write.insertInto("f1.driver_results")
SpanishGP.write.insertInto("f1.constructor_results")
```

5. Realizar un proceso automático en Airflow que orqueste los archivos creados en los puntos 3 y 4. Correrlo y mostrar una captura de pantalla (del DAG y del resultado en la base de datos)

```
from pyspark.context import SparkContext
from pyspark.sql.session import SparkSession
from pyspark.sql import HiveContext

# Configuramos...
sc = SparkContext('local')
spark = SparkSession(sc)
hc = HiveContext(sc)

# Traemos tablas
df1 = spark.read.option("header", "true").csv("hdfs://172.17.0.2:9000/ingest/results.csv")
df2 = spark.read.option("header", "true").csv("hdfs://172.17.0.2:9000/ingest/drivers.csv")
df3 = spark.read.option("header", "true").csv("hdfs://172.17.0.2:9000/ingest/constructors.csv")
df4 = spark.read.option("header", "true").csv("hdfs://172.17.0.2:9000/ingest/races.csv")

# Preparamos la primera transformacion
df1.createOrReplaceTempView("results")
df2.createOrReplaceTempView("drivers")

driver_results = spark.sql("""
    SELECT
        CAST(d.forename AS STRING) AS driver_forename,
        CAST(d.surname AS STRING) AS driver_surname,
        CAST(d.nationality AS STRING) AS driver_nationality,
        CAST(r.points AS INT) AS points
    FROM drivers d
    INNER JOIN results r ON d.driverId = r.driverId
    WHERE r.points!= 0
        """)


# Preparamos la segunda transformacion
df3.createOrReplaceTempView("constructors")
df4.createOrReplaceTempView("races")

SpanishGP = spark.sql("""
    SELECT
        CAST(c.constructorRef AS STRING) AS constructorref,
        CAST(c.name AS STRING) AS cons_name,
        CAST(c.nationality AS STRING) AS cons_nationality,
        CAST(c.url AS STRING) AS url,
        CAST(r.points AS INT) AS points
    FROM constructors c
    INNER JOIN results r ON c.constructorId = r.constructorId
    INNER JOIN  races ra ON ra.raceId = r.raceId
    WHERE ra.circuitID IN (4, 12, 26, 45, 49, 67) AND r.points != 0 AND ra.year = 1991
        """)

SpanishGP.show(3)

# Insertamos las tablas
driver_results.write.insertInto("f1.driver_results")
SpanishGP.write.insertInto("f1.constructor_results")
```



DAG: clase8

success | Schedule: 0 0 * * * | Next Run: 2024-05-27

Grid | Graph | Calendar | Task Duration | Task Tries | Landing Times | Gantt | Details | Code | Audit Log

2024-05-27T00:33:08Z | Runs 25 | Run manual__2024-05-27T00:33:07.078351+00:00 | Layout Left > Right | Find Task...

Update

BashOperator | DummyOperator

queued | running | success | failed | up_for_retry | up_for_reschedule | upstream_failed | skipped | scheduled | deferre

Auto-refres

comienza_proceso → ingest → transform → finaliza_proceso

queued | running | success | failed | up_for_retry | up_for_reschedule | upstream_failed | skipped | scheduled | deferred | no_status

Auto-refresh | Hide Det

comienza_proceso
ingest
transform
finaliza_proceso

DAG
clase8

DAG Details

DAG Runs Summary	
Total Runs Displayed	11
🟩 Total success	4
🟥 Total failed	7
First Run Start	2024-05-25, 03:55:56 UTC
Last Run Start	2024-05-27, 00:33:16 UTC
Max Run Duration	00:01:59
Mean Run Duration	00:00:44
Min Run Duration	00:00:16