

Clase 7

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

Diccionario de datos:

https://www.nyc.gov/assets/tlc/downloads/pdf/data_dictionary_trip_records_yellow.pdf

1. En Hive, crear la siguiente tabla (externa) en la base de datos tripdata: a.
airport_trips(tpep_pickup_datetime, airport_fee, payment_type, tolls_amount, total_amount)

Muestra la tablas actuales dentro de la base tripdata:

```
hive> show tables;
OK
tripdata_table
tripdata_table_km
Time taken: 0.027 seconds, Fetched: 2 row(s)
```

```
create external table tripdata.airport_trips(tpep_pickup_datetime timestamp, airport_fee
double, payment_type bigint, tolls_amount double, total_amount double)
> comment 'tabla externa airport trips clase 7'
> stored as parquet
> location '/tables/airport_trips';
```

Con respecto a payment_type **bigint**, hive no aceptó el tipo de dato "long" como estaba en el parquet, asique la IA me recomendó **bigint**. Y a los demás campos le puse el tipo de dato con el que están el archivo parquet.

```
hive> create external table tripdata.airport_trips(tpep_pickup_datetime timestamp, airport_f
ee double, payment_type bigint, tolls_amount double, total_amount double)
> comment 'tabla externa airport trips clase 7'
> stored as parquet
> location '/tables/airport_trips';
OK
Time taken: 0.039 seconds
```

```
hadoop@5f1a3da9dadf:/$ hdfs dfs -ls /tables/
Found 1 items
drwxr-xr-x - hadoop supergroup 0 2025-10-14 14:27 /tables/airport_trips
```

2. En Hive, mostrar el esquema de airport_trips

```
hive> describe airport_trips;
OK
tpep_pickup_datetime    timestamp
airport_fee             double
payment_type            bigint
tolls_amount            double
total_amount            double
Time taken: 0.028 seconds, Fetched: 5 row(s)
```

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

Yellow_tripdata_2021-01.parquet:

https://dataengineerpublic.blob.core.windows.net/data-engineer/yellow_tripdata_2021-01.parquet

Yellow_tripdata_2021-02.parquet:

https://dataengineerpublic.blob.core.windows.net/data-engineer/yellow_tripdata_2021-02.parquet

Script:

```
#Descarga de archivo 2021-01 del sitio
https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page
wget -P /home/hadoop/landing
https://d37ci6vzurychx.cloudfront.net/trip-data/yellow_tripdata_2021-01.parquet
```

```
#Descarga de archivo 2021-02 del sitio
https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page
wget -P /home/hadoop/landing
https://d37ci6vzurychx.cloudfront.net/trip-data/yellow_tripdata_2021-02.parquet
```

```
#Copia de archivos desde carpeta Landing al HDFS.
/home/hadoop/hadoop/bin/hdfs dfs -put
/home/hadoop/landing/yellow_tripdata_2021-01.parquet /ingest
```

```
/home/hadoop/hadoop/bin/hdfs dfs -put
/home/hadoop/landing/yellow_tripdata_2021-02.parquet /ingest
```

```
#Borrado de archivo 2021-01 de carpeta Landing
rm /home/hadoop/landing/yellow_tripdata_2021-01.parquet
```

```
#Borrado de archivo 2021-02 de carpeta Landing
```

```
rm /home/hadoop/landing/yellow_tripdata_2021-02.parquet
```

```
hadoop@5f1a3da9dadf:~/scripts$ cat > ingest_clase7.sh
#Descarga de archivo 2021-01 del sitio https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page
wget -P /home/hadoop/landing https://d37ci6vzurychx.cloudfront.net/trip-data/yellow_tripdata_2021-01.parquet

#Descarga de archivo 2021-02 del sitio https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page
wget -P /home/hadoop/landing https://d37ci6vzurychx.cloudfront.net/trip-data/yellow_tripdata_2021-02.parquet

#Copia de archivos desde carpeta Landing al HDFS.
hdfs dfs -put /home/hadoop/landing/yellow_tripdata_2021-01.parquet /ingest
hdfs dfs -put /home/hadoop/landing/yellow_tripdata_2021-02.parquet /ingest

#Borrado de archivo 2021-01 de carpeta Landing
rm /home/hadoop/landing/yellow_tripdata_2021-01.parquet

#Borrado de archivo 2021-02 de carpeta Landing
rm /home/hadoop/landing/yellow_tripdata_2021-02.parquet hadoop@5f1a3da9dadf:~/script
```

4. Crear un archivo .py que permita, mediante Spark, crear un data frame uniendo los viajes del mes 01 y mes 02 del año 2021 y luego Insertar en la tabla airport_trips los viajes que tuvieron como inicio o destino aeropuertos, que hayan pagado con dinero.

Leyendo la user guide en <https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page>

Data Dictionaries and MetaData

- [Trip Record User Guide](#)

PULocationID & DOLocationID—matching zone numbers to the map

Each of the trip records contains a field corresponding to the location of the pickup or drop-off of the trip (or in FHV records before 2017, just the pickup), populated by numbers ranging from 1-263. These numbers correspond to taxi zones, which may be downloaded as a table or map/shapefile and matched to the trip records using a join. The data is currently available on the Open Data Portal at <https://data.cityofnewyork.us/Transportation/NYC-Taxi-Zones/d3c5-ddgc>, or on the trip records page on the TLC website, <https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page>, under Taxi Zone Maps and Lookup Tables, see below:

Estos 2 campos PULocationID y DOLocationID muestran el pickup y drop-off, entiendo

que correspondería a **inicio** y **destino**. Los LocationID correspondientes a aeropuertos:

Taxi Zone Maps and Lookup Tables

- **Taxi Zone Lookup Table** (CSV)
- **Taxi Zone Shapefile** (PARQUET)

A	B	C	D
LocationID	Borough	Zone	service_zone
1	EWB	Newark Airport	EWB
132	Queens	JFK Airport	Airports
138	Queens	LaGuardia Airport	Airports

Haciendo un analisis previo primero y uniendo los 2 archivos no encontré nulos para **PULocationID** y **DOLocationID** asique entiendo los puedo utilizar como filtro sin otra operación previa para una consulta sql.

```
>>> df_01 = spark.read.parquet("hdfs://172.17.0.2:9000/ingest/yellow_tripdata_2021-01.parquet")
>>> df_02 = spark.read.parquet("hdfs://172.17.0.2:9000/ingest/yellow_tripdata_2021-02.parquet")
>>> df_union = df_01.union(df_02)
>>> df_union.createOrReplaceTempView("airport_vista")
>>> new_df = spark.sql("select * from airport_vista where PULocationID is null")
>>> new_df.count()
0
>>> new_df = spark.sql("select * from airport_vista where DOLocationID is null")
>>> new_df.count()
0
```

Además veo que muchos valores en el campo **airport_fee** son null, pero entiendo que no importa porque no los vamos a utilizar para una operación de suma por ejemplo, en este caso es simplemente para contar cantidad de viajes.

Por lo último, la IA me recomendó trabajar con los archivos parquet por separado por temas de performance y después unirlos ya transformados.

Entonces el archivo de transformación **t_clase7.py** queda así:

```
from pyspark.context import SparkContext
from pyspark.sql.session import SparkSession
sc = SparkContext('local')
spark = SparkSession(sc)
```

```
from pyspark.sql import HiveContext
hc = HiveContext(sc)
```

```

##Leyendo los 2 archivos parquet mes 01 y 02 de carpeta /ingest
df_01 =
    spark.read.parquet("hdfs://172.17.0.2:9000/ingest/yellow_tripdata_2021-01.parquet")
df_02 =
    spark.read.parquet("hdfs://172.17.0.2:9000/ingest/yellow_tripdata_2021-02.parquet")

#Crea las vistas de cada parquet
df_01.createOrReplaceTempView("airport_vista_1")
df_02.createOrReplaceTempView("airport_vista_2")

#Filtro columnas y por viajes que se hayan pagado cash y que tengan como inicio ó
    destino aeropuertos

df_01_filtro = spark.sql("select cast(tpep_pickup_datetime as timestamp),
    cast(airport_fee as double), cast(payment_type as bigint), cast(tolls_amount as
    double), cast(total_amount as double) from airport_vista_1 where payment_type =
    2 and (PULocationID IN (1, 132, 138) or DOLocationID in (1, 132, 138)) ")

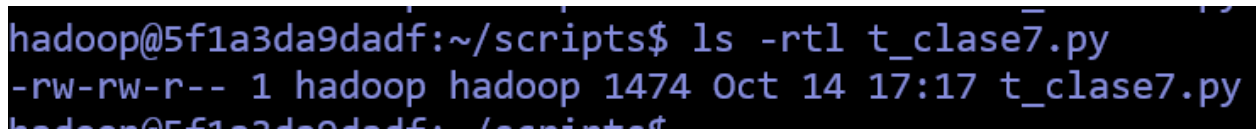
df_02_filtro = spark.sql("select cast(tpep_pickup_datetime as timestamp),
    cast(airport_fee as double), cast(payment_type as bigint), cast(tolls_amount as
    double), cast(total_amount as double) from airport_vista_2 where payment_type =
    2 and (PULocationID IN (1, 132, 138) or DOLocationID in (1, 132, 138)) ")

#Uno los dataframes
df_air_unido = df_01_filtro.union(df_02_filtro)

#Hago la vista para insertar
df_air_unido.createOrReplaceTempView("airport_trips_insert")

#Hace load en HIVE
hc.sql("insert into tripdata.airport_trips select * from airport_trips_insert")

```



```

hadoop@5f1a3da9dadf:~/scripts$ ls -rtl t_clase7.py
-rw-rw-r-- 1 hadoop hadoop 1474 Oct 14 17:17 t_clase7.py
hadoop@5f1a3da9dadf:~/scripts$

```

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)

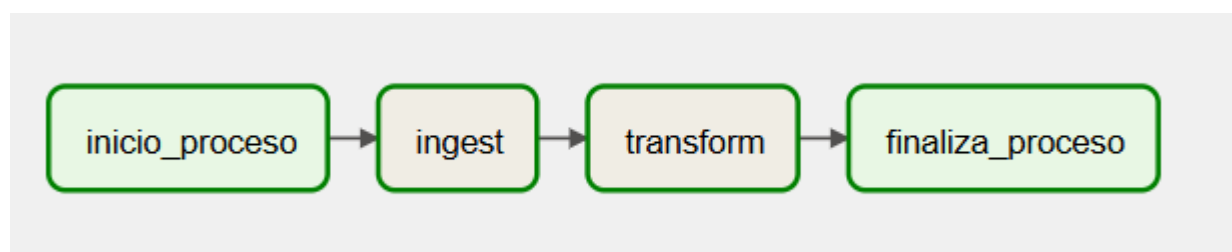
```

dag.cli()hadoop@5f1a3da9dadf:~/airflow/dags$ ls -rtl
total 20
-rw-rw-r-- 1 hadoop hadoop 1079 May  1  2022 example-DAG.py
-rw-rw-r-- 1 hadoop hadoop 1024 May  5  2022 ingest-transform.py
-rw-rw-r-- 1 hadoop hadoop 1028 Oct 11 15:04 ingest-starwars.py
drwxrwxr-x 1 hadoop hadoop 4096 Oct 11 15:04 __pycache__
-rw-rw-r-- 1 hadoop hadoop 1039 Oct 14 16:32 ingest_clase7.py
hadoop@5f1a3da9dadf:~/airflow/dags$

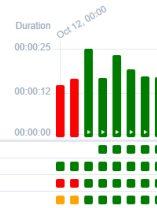
```

DAGs

All 4		Active 0	Paused 4	✕ ingest	
<i>i</i>	DAG ↕	Owner ↕	Runs <i>i</i>	Schedule	
<input type="checkbox"/>	example-DAG ingest transform	airflow	<div><div></div><div></div><div></div><div></div></div>	00*** <i>i</i>	
<input type="checkbox"/>	ingest-starwars ingest transform	airflow	<div><div></div><div>1</div><div></div><div>1</div></div>	00*** <i>i</i>	
<input type="checkbox"/>	ingest-transform ingest transform	airflow	<div><div></div><div>7</div><div></div><div></div></div>	00*** <i>i</i>	
<input type="checkbox"/>	ingest_clase7 ingest transform	airflow	<div><div></div><div></div><div></div><div></div></div>	00*** <i>i</i>	



inicio_proceso
ingest
transform
finaliza_proceso



DAG
ingest_clase7

DAG Details

DAG Runs Summary

Total Runs Displayed	8
Total success	6
Total failed	2
First Run Start	2025-10-14, 19:41:52 UTC
Last Run Start	2025-10-14, 20:36:01 UTC

```
hive> select count(*) from airport_trips;
```

```
Total MapReduce CPU Time Spent: 0 msec  
OK  
30611  
Time taken: 2.336 seconds, Fetched: 1 row(s)  
hive>
```

```
hive> SET hive.cli.print.header=true;  
hive> select * from airport_trips limit 20;  
OK  
airport_trips.tpep_pickup_datetime    airport_trips.airport_fee    airport_trips.payment_type    airport_trips.tolls_amounta  
airport_trips.total_amount  
2020-12-31 21:10:46    NULL    2    6.12    33.92  
2020-12-31 21:37:40    NULL    2    6.12    59.42  
2020-12-31 21:24:30    NULL    2    0.0    40.3  
2020-12-31 21:06:08    NULL    2    0.0    13.3  
2020-12-31 21:38:23    NULL    2    0.0    20.3  
2020-12-31 21:33:51    NULL    2    0.0    47.3
```