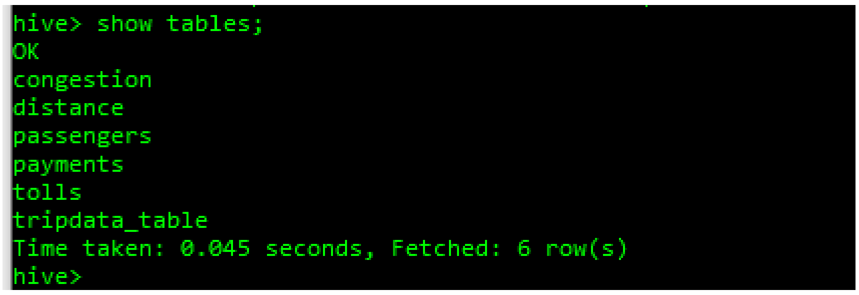
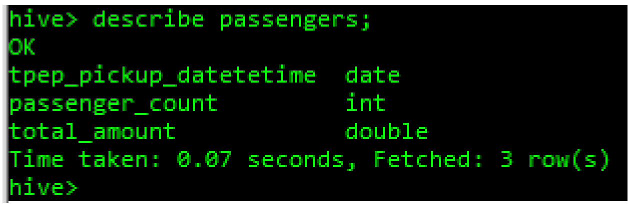
1. En Hive, crear las siguientes tablas (internas) en la base de datos tripdata en hive:

```
CREATE TABLE tripdata.payments(VendorID int, tpep_pickup_datetetime date, payment_type
int, total_amount double)
COMMENT 'Payments table'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';

CREATE TABLE tripdata.passengers(tpep_pickup_datetetime date, passenger_count int,
total_amount double)
COMMENT 'passengers table'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';

CREATE TABLE tripdata.tolls(tpep_pickup_datetetime date, passenger_count int, tolls_amount
double, total_amount double)
COMMENT 'tolls table'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';

CREATE TABLE tripdata.congestion(tpep_pickup_datetetime date, passenger_count int,
congestion_surcharge double, total_amount double)
COMMENT 'congestion table'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';

CREATE TABLE tripdata.distance(tpep_pickup_datetetime date, passenger_count int,
trip_distance double, total_amount double)
COMMENT 'distance table'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';
```

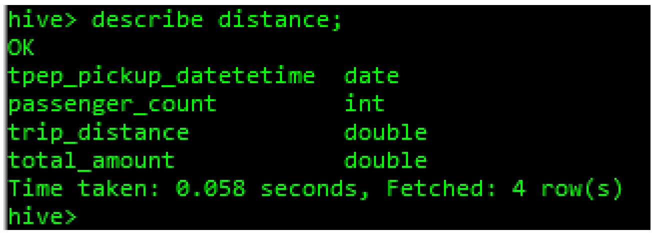


```
hive> show tables;
OK
congestion
distance
passengers
payments
tolls
tripdata_table
Time taken: 0.045 seconds, Fetched: 6 row(s)
hive>
```

2. En Hive, hacer un ‘describe’ de las tablas passengers y distance.



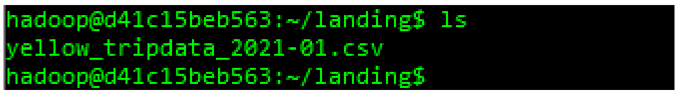
```
hive> describe passengers;
OK
tpep_pickup_datetetime  date
passenger_count         int
total_amount            double
Time taken: 0.07 seconds, Fetched: 3 row(s)
hive>
```



```
hive> describe distance;
OK
tpep_pickup_datetetime  date
passenger_count         int
trip_distance           double
total_amount            double
Time taken: 0.058 seconds, Fetched: 4 row(s)
hive>
```

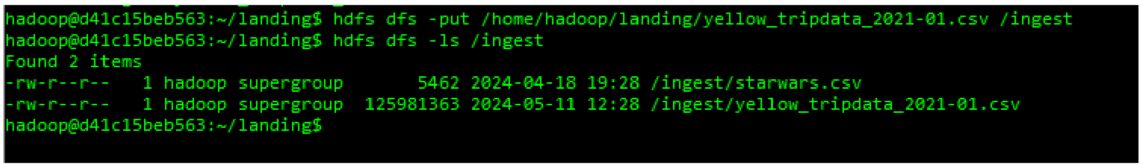
3. Hacer ingest del file: Yellow_tripodata_2021-01.csv.

wget -P /home/hadoop/landing https://dataengineerpublic.blob.core.windows.net/data-engineer/yellow_tripdata_2021-01.csv



```
hadoop@d41c15beb563:~/landing$ ls
yellow_tripdata_2021-01.csv
hadoop@d41c15beb563:~/landing$
```

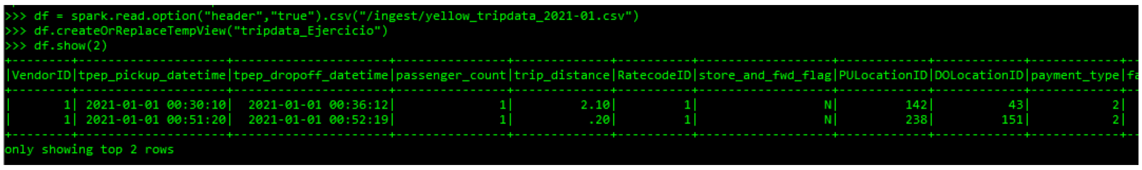
hdfs dfs -put /home/hadoop/landing/yellow_tripdata_2021-01.csv /ingest



```
hadoop@d41c15beb563:~/landing$ hdfs dfs -put /home/hadoop/landing/yellow_tripdata_2021-01.csv /ingest
hadoop@d41c15beb563:~/landing$ hdfs dfs -ls /ingest
Found 2 items
-rw-r--r--   1 hadoop supergroup       5462 2024-04-18 19:28 /ingest/starwars.csv
-rw-r--r--   1 hadoop supergroup  125981363 2024-05-11 12:28 /ingest/yellow_tripdata_2021-01.csv
hadoop@d41c15beb563:~/landing$
```

Para los siguientes ejercicios, debes usar PySpark (obligatorio). Si deseas practicar más, también puedes repetir los mismos en SQL (opcional)

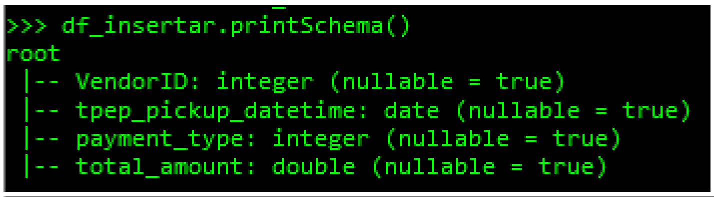
4. (Opcional SQL) Generar una vista
df.createOrReplaceTempView("tripdata_Ejercicio")



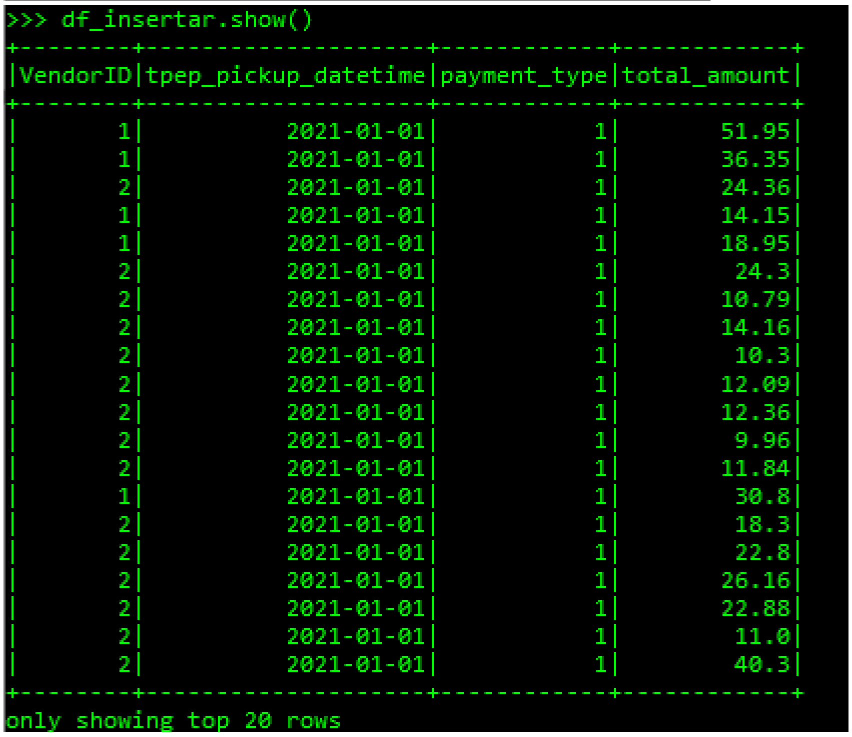
```
>>> df = spark.read.option("header","true").csv("/ingest/yellow_tripdata_2021-01.csv")
>>> df.createOrReplaceTempView("tripdata_Ejercicio")
>>> df.show(2)
+--------+--------------------+---------------------+---------------+-------------+----------+------------------+------------+------------+------------+-+
|VendorID|tpep_pickup_datetime|tpep_dropoff_datetime|passenger_count|trip_distance|RatecodeID|store_and_fwd_flag|PULocationID|DOLocationID|payment_type|fa
+--------+--------------------+---------------------+---------------+-------------+----------+------------------+------------+------------+------------+-+
|       1| 2021-01-01 00:30:10|  2021-01-01 00:36:12|              1|         2.10|         1|                 N|         142|          43|           2|
|       1| 2021-01-01 00:51:20|  2021-01-01 00:52:19|              1|          .20|         1|                 N|         238|         151|           2|
+--------+--------------------+---------------------+---------------+-------------+----------+------------------+------------+------------+------------+-+
only showing top 2 rows
```

5.Insertar en la tabla payments (VendorID, tpep_pickup_datetetime, payment_type, total_amount) Solamente los pagos con tarjeta de crédito

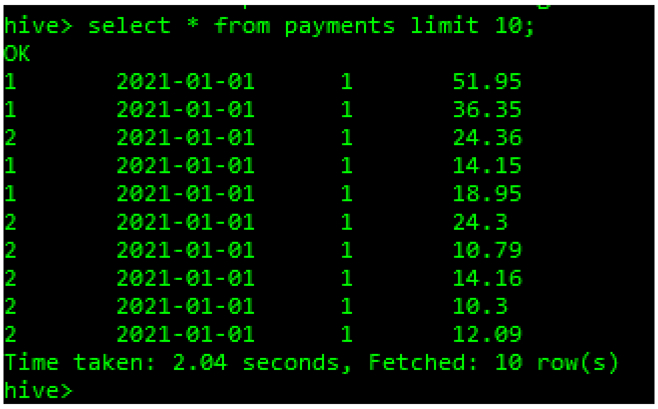
- df_filtrar = df.filter((df.payment_type == 1))
- df_insertar = df_filtrar.select(df_filtrar.VendorID.cast("int"), df_filtrar.tpep_pickup_datetime.cast("date"), df_filtrar.payment_type.cast("int"), df_filtrar.total_amount.cast("double"))
- df_insertar.write.insertInto("tripdata.payments")



```
>>> df_insertar.printSchema()
root
 |-- VendorID: integer (nullable = true)
 |-- tpep_pickup_datetime: date (nullable = true)
 |-- payment_type: integer (nullable = true)
 |-- total_amount: double (nullable = true)
```



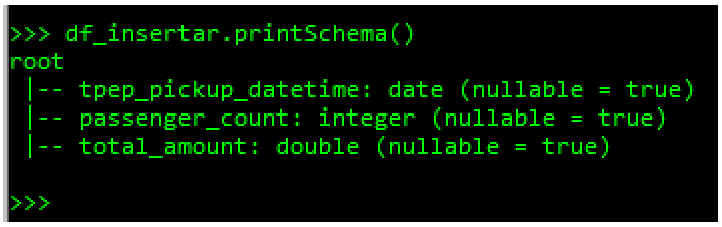
```
>>> df_insertar.show()
+--------+--------------------+------------+------------+
|VendorID|tpep_pickup_datetime|payment_type|total_amount|
+--------+--------------------+------------+------------+
|       1|          2021-01-01|           1|       51.95|
|       1|          2021-01-01|           1|       36.35|
|       2|          2021-01-01|           1|       24.36|
|       1|          2021-01-01|           1|       14.15|
|       1|          2021-01-01|           1|       18.95|
|       2|          2021-01-01|           1|        24.3|
|       2|          2021-01-01|           1|       10.79|
|       2|          2021-01-01|           1|       14.16|
|       2|          2021-01-01|           1|        10.3|
|       2|          2021-01-01|           1|       12.09|
|       2|          2021-01-01|           1|       12.36|
|       2|          2021-01-01|           1|        9.96|
|       2|          2021-01-01|           1|       11.84|
|       1|          2021-01-01|           1|        30.8|
|       2|          2021-01-01|           1|        18.3|
|       2|          2021-01-01|           1|        22.8|
|       2|          2021-01-01|           1|       26.16|
|       2|          2021-01-01|           1|       22.88|
|       2|          2021-01-01|           1|        11.0|
|       2|          2021-01-01|           1|        40.3|
+--------+--------------------+------------+------------+
only showing top 20 rows
```



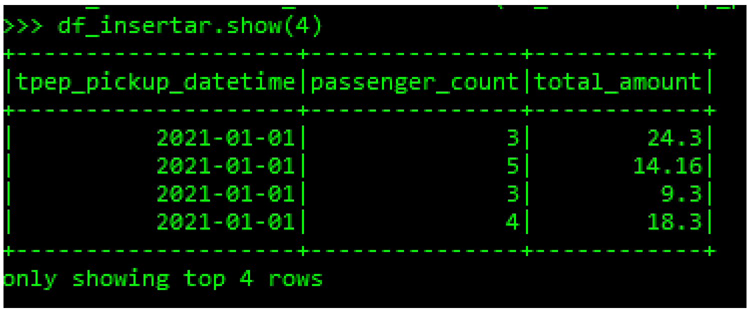
```
hive> select * from payments limit 10;
OK
1       2021-01-01      1       51.95
1       2021-01-01      1       36.35
2       2021-01-01      1       24.36
1       2021-01-01      1       14.15
1       2021-01-01      1       18.95
2       2021-01-01      1       24.3
2       2021-01-01      1       10.79
2       2021-01-01      1       14.16
2       2021-01-01      1       10.3
2       2021-01-01      1       12.09
Time taken: 2.04 seconds, Fetched: 10 row(s)
hive>
```

6. Insertar en la tabla passengers (tpep_pickup_datetetime, passenger_count, total_amount) los registros cuya cantidad de pasajeros sea mayor a 2 y el total del viaje cueste más de 8 dólares

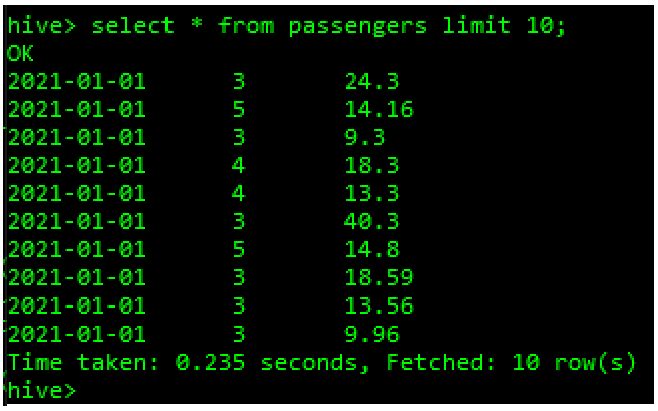
- df_filtrar = df.filter((df.passenger_count > 2) & (df.total_amount > 8))
- df_insertar = df_filtrar.select(df_filtrar.tpep_pickup_datetime.cast("date"), df_filtrar.passenger_count.cast("int"), df_filtrar.total_amount.cast("double"))
- df_insertar.write.insertInto("tripdata.passengers")



```
>>> df_insertar.printSchema()
root
 |-- tpep_pickup_datetime: date (nullable = true)
 |-- passenger_count: integer (nullable = true)
 |-- total_amount: double (nullable = true)

>>>
```



```
>>> df_insertar.show(4)
+--------------------+---------------+------------+
|tpep_pickup_datetime|passenger_count|total_amount|
+--------------------+---------------+------------+
|          2021-01-01|              3|        24.3|
|          2021-01-01|              5|       14.16|
|          2021-01-01|              3|         9.3|
|          2021-01-01|              4|        18.3|
+--------------------+---------------+------------+
only showing top 4 rows
```

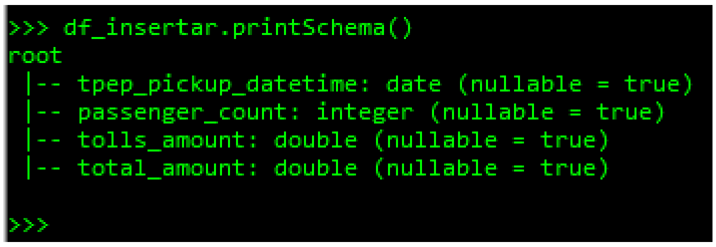


```
hive> select * from passengers limit 10;
OK
2021-01-01      3       24.3
2021-01-01      5       14.16
2021-01-01      3       9.3
2021-01-01      4       18.3
2021-01-01      4       13.3
2021-01-01      3       40.3
2021-01-01      5       14.8
2021-01-01      3       18.59
2021-01-01      3       13.56
2021-01-01      3       9.96
Time taken: 0.235 seconds, Fetched: 10 row(s)
hive>
```

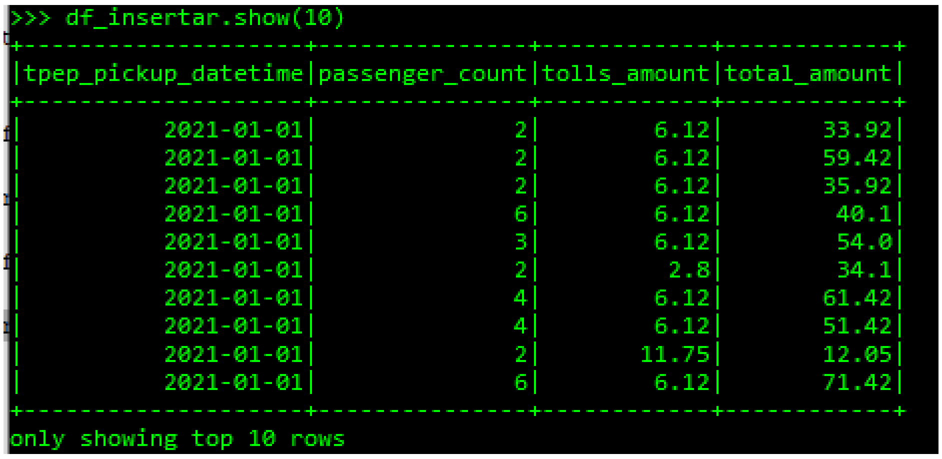
7. Insertar en la tabla tolls (tpep_pickup_datetetime, passenger_count, tolls_amount, total_amount) los registros que tengan pago de peajes mayores a 0.1 y cantidad de pasajeros mayores a 1.

- dffilter = df.filter((df.passenger_count > 1) & (df.tolls_amount > 0.1))
- df_insertar = df_filtrar.select(df_filtrar.tpep_pickup_datetime.cast("date"), df_filtrar.passenger_count.cast("int"), df_filtrar.tolls_amount.cast("double"), df_filtrar.total_amount.cast("double"))

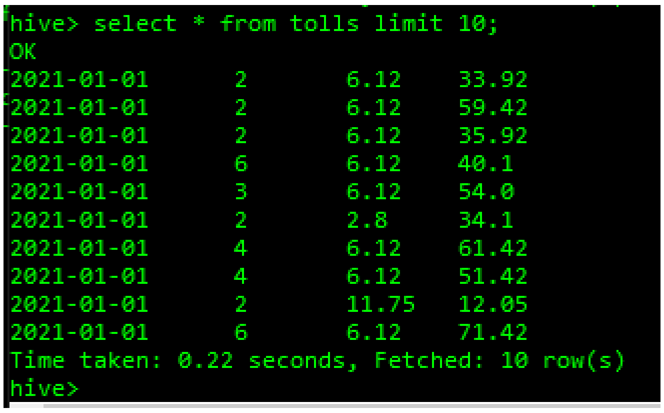
- df_insertar.write.insertInto("tripdata.tolls")



```
>>> df_insertar.printSchema()
root
 |-- tpep_pickup_datetime: date (nullable = true)
 |-- passenger_count: integer (nullable = true)
 |-- tolls_amount: double (nullable = true)
 |-- total_amount: double (nullable = true)

>>>
```



```
>>> df_insertar.show(10)
+--------------------+---------------+------------+------------+
|tpep_pickup_datetime|passenger_count|tolls_amount|total_amount|
+--------------------+---------------+------------+------------+
|          2021-01-01|              2|        6.12|       33.92|
|          2021-01-01|              2|        6.12|       59.42|
|          2021-01-01|              2|        6.12|       35.92|
|          2021-01-01|              6|        6.12|        40.1|
|          2021-01-01|              3|        6.12|        54.0|
|          2021-01-01|              2|         2.8|        34.1|
|          2021-01-01|              4|        6.12|       61.42|
|          2021-01-01|              4|        6.12|       51.42|
|          2021-01-01|              2|       11.75|       12.05|
|          2021-01-01|              6|        6.12|       71.42|
+--------------------+---------------+------------+------------+
only showing top 10 rows
```



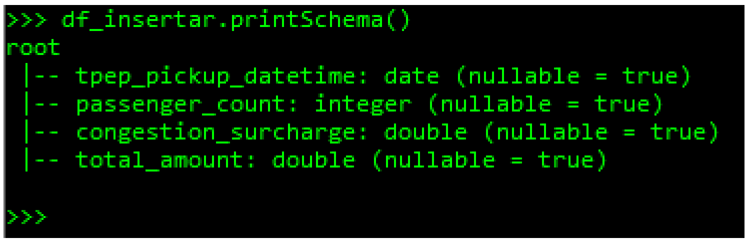
```
hive> select * from tolls limit 10;
OK
2021-01-01      2       6.12    33.92
2021-01-01      2       6.12    59.42
2021-01-01      2       6.12    35.92
2021-01-01      6       6.12    40.1
2021-01-01      3       6.12    54.0
2021-01-01      2       2.8     34.1
2021-01-01      4       6.12    61.42
2021-01-01      4       6.12    51.42
2021-01-01      2       11.75   12.05
2021-01-01      6       6.12    71.42
Time taken: 0.22 seconds, Fetched: 10 row(s)
hive>
```

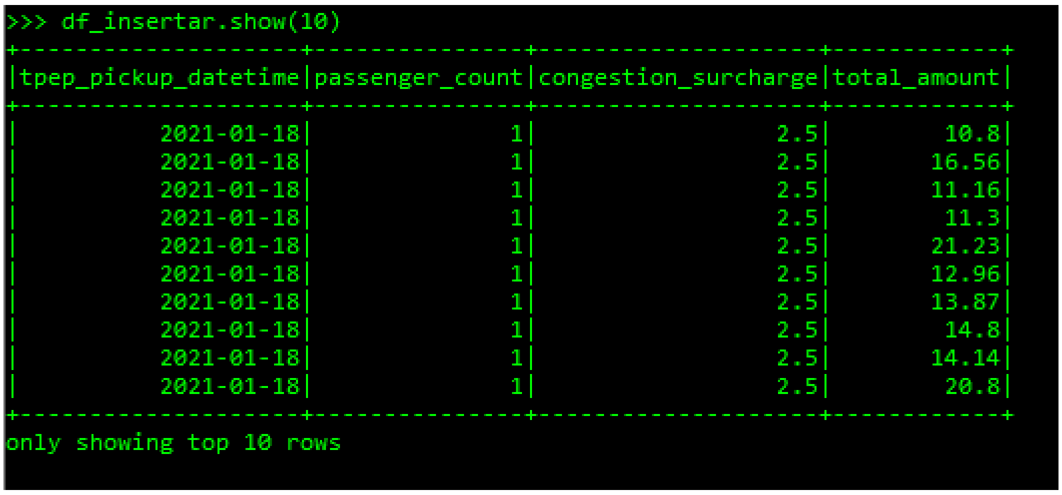
8. Insertar en la tabla congestion (tpep_pickup_datetetime, passenger_count, congestion_surcharge, total_amount) los registros que hayan tenido congestión en los viajes en la fecha 2021-01-18

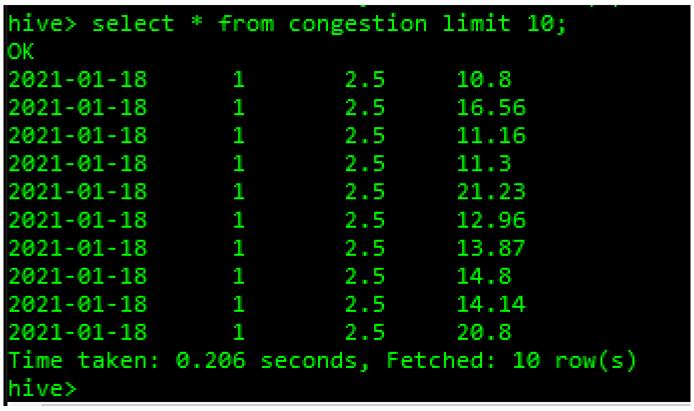
- df_filtrar = df.filter((df.tpep_pickup_datetime.cast("date") == "2021-01-18") & (df.congestion_surcharge > 0))

- df_insertar = df_filtrar.select(df_filtrar.tpep_pickup_datetime.cast("date"), df_filtrar.passenger_count.cast("int"), df_filtrar.congestion_surcharge.cast("double"), df_filtrar.total_amount.cast("double"))

- df_insertar.write.insertInto("tripdata.congestion")


```
>>> df_insertar.printSchema()
root
 |-- tpep_pickup_datetime: date (nullable = true)
 |-- passenger_count: integer (nullable = true)
 |-- congestion_surcharge: double (nullable = true)
 |-- total_amount: double (nullable = true)

>>>
```


```
>>> df_insertar.show(10)
+--------------------+---------------+--------------------+------------+
|tpep_pickup_datetime|passenger_count|congestion_surcharge|total_amount|
+--------------------+---------------+--------------------+------------+
|          2021-01-18|              1|                 2.5|        10.8|
|          2021-01-18|              1|                 2.5|       16.56|
|          2021-01-18|              1|                 2.5|       11.16|
|          2021-01-18|              1|                 2.5|        11.3|
|          2021-01-18|              1|                 2.5|       21.23|
|          2021-01-18|              1|                 2.5|       12.96|
|          2021-01-18|              1|                 2.5|       13.87|
|          2021-01-18|              1|                 2.5|        14.8|
|          2021-01-18|              1|                 2.5|       14.14|
|          2021-01-18|              1|                 2.5|        20.8|
+--------------------+---------------+--------------------+------------+
only showing top 10 rows
```

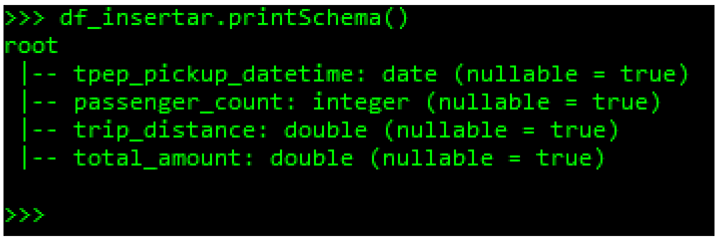

```
hive> select * from congestion limit 10;
OK
2021-01-18      1       2.5     10.8
2021-01-18      1       2.5     16.56
2021-01-18      1       2.5     11.16
2021-01-18      1       2.5     11.3
2021-01-18      1       2.5     21.23
2021-01-18      1       2.5     12.96
2021-01-18      1       2.5     13.87
2021-01-18      1       2.5     14.8
2021-01-18      1       2.5     14.14
2021-01-18      1       2.5     20.8
Time taken: 0.206 seconds, Fetched: 10 row(s)
hive>
```

9. Insertar en la tabla distance (tpep_pickup_datetetime, passenger_count, trip_distance, total_amount) los registros de la fecha 2020-12-31 que hayan tenido solamente un pasajero (passenger_count = 1) y hayan recorrido más de 15 millas (trip_distance).

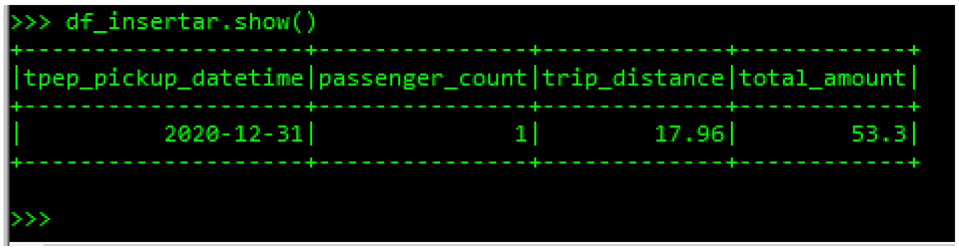
- df_filtrar = df.filter((df.tpep_pickup_datetime.cast("date") == "2020-12-31") & (df.passenger_count == 1) & (df.trip_distance > 15))

- df_insertar = df_filtrar.select(df_filtrar.tpep_pickup_datetime.cast("date"), df_filtrar.passenger_count.cast("int"), df_filtrar.trip_distance.cast("double"), df_filtrar.total_amount.cast("double"))

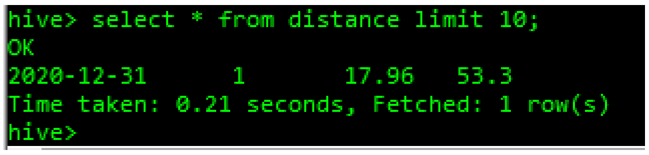
- df_insertar.write.insertInto("tripdata.distance")



```
>>> df_insertar.printSchema()
root
 |-- tpep_pickup_datetime: date (nullable = true)
 |-- passenger_count: integer (nullable = true)
 |-- trip_distance: double (nullable = true)
 |-- total_amount: double (nullable = true)

>>>
```



```
>>> df_insertar.show()
+--------------------+---------------+-------------+------------+
|tpep_pickup_datetime|passenger_count|trip_distance|total_amount|
+--------------------+---------------+-------------+------------+
|          2020-12-31|              1|        17.96|        53.3|
+--------------------+---------------+-------------+------------+

>>>
```



```
hive> select * from distance limit 10;
OK
2020-12-31      1       17.96   53.3
Time taken: 0.21 seconds, Fetched: 1 row(s)
hive>
```