# dsti-a19/ajourdan/lab2

READY

# Spark Dataframe Lab: NYC Taxi dataset

More info on the dataset: https://training.ververica.com/setup/taxiData.html (https://training.ververica.com/setup/taxiData.html)

FINISHED

```
%pyspark
from pyspark.sql.types import *

# Define the schema of the nycTaxiFares dataset
fares_schema = StructType([
    StructField('ride_id', IntegerType(), False),
    StructField('taxi_id', IntegerType(), False),
    StructField('driver_id', IntegerType(), False),
    StructField('start_time', TimestampType(), False),
    StructField('payment_type', StringType(), False),
    StructField('tip', FloatType(), False),
    StructField('tolls', FloatType(), False),
    StructField('total_fare', FloatType(), False)
])

# Create a DataFrame from the nycTaxiFares.gz file
taxi_fares = spark.read.csv(
    'hdfs:///learning/data/nycTaxi/nycTaxiFares.csv',
    schema=fares_schema)

# Get an idea of the dataset
# taxi_fares.show(10)
```

Took 0 sec. Last updated by a.jourdan-dsti at March 17 2020, 9:48:54 PM.

FINISHED

```
%pyspark
taxi_fares.count()
```

```
1499999
```

Took 1 sec. Last updated by a.jourdan-dsti at March 17 2020, 9:47:03 PM.

FINISHED

```
%pyspark

from pyspark.sql.functions import *

# Gather information for each driver
driver_summary = taxi_fares \
    .groupBy('driver_id') \
    .agg(
        count('ride_id').alias('rides_nb'),
        sum('tip').alias('tip_total'),
        sum('tolls').alias('tolls_total'),
        sum('total_fare').alias('fares_total')
    )

#driver_summary.show(10)
```

Took 0 sec. Last updated by a.jourdan-dsti at March 17 2020, 9:49:06 PM.

FINISHED

```
%pyspark
# Compute driver's performance

driver_perf = driver_summary \
    .withColumn('avg_tip', col('tip_total')/col('rides_nb')) \
    .withColumn('avg_tolls', col('tolls_total')/col('rides_nb')) \
    .withColumn('avg_fare', col('fares_total')/col('rides_nb')) \
    .withColumn('avg_fare_without_tolls', (col('fares_total')-col('tolls_total'))/col('rides_nb'

#driver_perf.show(10)
```

Took 0 sec. Last updated by a.jourdan-dsti at March 17 2020, 9:49:13 PM.

FINISHED

```
%pyspark
taxi_rides = spark.read.csv(
    'hdfs:///education/ece/spark/labs/2/nycTaxiRides.gz')

#taxi_rides.show(10)
```

Took 0 sec. Last updated by a.jourdan-dsti at March 17 2020, 9:49:20 PM.

FINISHED

```
%pyspark
rides_schema = StructType([
    StructField('ride_id', IntegerType(), False),
    StructField('is_start', StringType(), False),
    StructField('end_time', TimestampType(), False),
    StructField('start_time', TimestampType(), False),
    StructField('start_lon', FloatType(), False),
    StructField('start_lat', FloatType(), False),
    StructField('end_lon', FloatType(), False),
    StructField('end_lat', FloatType(), False),
    StructField('passenger_count', IntegerType(), False),
    StructField('taxi_id', IntegerType(), False),
    StructField('driver_id', IntegerType(), False)
])

taxi_rides = spark.read.csv(
    'hdfs:///education/ece/spark/labs/2/nycTaxiRides.gz',
    schema=rides_schema)

#taxi_rides.show(10)
```

Took 0 sec. Last updated by a.jourdan-dsti at March 17 2020, 9:49:27 PM.

FINISHED

```
%pyspark
# Filter the rides dataset to keep only END events
taxi_rides_end = taxi_rides.where('is_start = "END"').drop('taxi_id', 'driver_id', 'start_time')

#z.show(taxi_rides_end)
```

Took 0 sec. Last updated by a.jourdan-dsti at March 17 2020, 9:49:33 PM.

FINISHED

```
%pyspark
# Join the 2 DataFrames
complete_rides = taxi_fares.join(taxi_rides_end, 'ride_id').drop('is_start')

#complete_rides.show(10)
```

Took 0 sec. Last updated by a.jourdan-dsti at March 17 2020, 9:49:39 PM.

FINISHED

# TODO

1. Computed the total money earned by each driver for each day
2. Compute for each:
    - The average tip
    - The average duration
    - The average “distance at the crow flies” (“à vol d’oiseau”) - optional
3. Find the hour of the day when the tips are the highest
4. Find the percentage of each type of payment
5. Find the top 10 drivers that:
    - Won the most money
    - Did the greater number of rides



Took 0 sec. Last updated by a.jourdan-dsti at March 17 2020, 3:05:49 PM.

**Computed the total money earned by each driver for each day** FINISHED

Took 0 sec. Last updated by a.jourdan-dsti at March 17 2020, 9:50:05 PM.

FINISHED

```
%spark2.pyspark
from pyspark.sql import functions as F

taxi_rides_end = taxi_rides.where('is_start = "END"').select('taxi_id', 'driver_id', 'end_time',
                                                    .withColumn('newDate',col('end_time').cast('
                                                    .drop('end_time')

# taxi_rides_end.show(10)

complete_rides = taxi_fares.join(taxi_rides_end, 'ride_id').select(taxi_fares.taxi_id, taxi_fare

driver_summary = complete_rides \
    .groupBy('driver_id','newDate') \
    .agg(
        F.count('ride_id').alias('rides_nb'),
        F.sum('tip').alias('tip_total'),
        F.sum('tolls').alias('tolls_total'),
        F.sum('total_fare').alias('fares_total')
    )

driver_tot_money = driver_summary \
    .withColumn('money_total', (col('fares_total')+col('tip_total')-col('tolls_total')))

driver_tot_money = driver_tot_money.orderBy(driver_tot_money.newDate.desc())

driver_tot_money.show(10)
```

```
+----------+----------+--------+------------------+-----------------+------------------+--------
----------+
| driver_id|   newDate|rides_nb|         tip_total|      tolls_total|       fares_total|       m
oney_total|
+----------+----------+--------+------------------+-----------------+------------------+--------
----------+
|2013015262|2013-01-24|       1|               0.0|4.800000190734863| 35.79999923706055|30.99999
9046325684|
|2013007591|2013-01-24|       1|3.1500000953674316|              0.0|13.649999618530273|16.79999
9713897705|
|2013025561|2013-01-11|       1|               1.5|              0.0|               9.0|
10.5|
|2013005706|2013-01-04|       6|11.179999947547913|              0.0|103.68000030517578| 114.860
```

```
0002527237|
|2013024970|2013-01-04|       2| 3.899999976158142|             0.0| 39.39999961853027|43.29999
9594688416|
|2013004476|2013-01-04|       5|13.880000114440918|             0.0|103.88000106811523|117.7600
```

# dsti-a19/ajourdan/lab2

Took 25 sec. Last updated by a.jourdan-dsti at March 17 2020, 3:57:29 PM.

FINISHED

```
%spark2.pyspark
from pyspark.sql import functions as F

df_fares = taxi_fares \
            .withColumn('newDate',taxi_fares.start_time.cast('Date')) \
            .drop('taxi_id','start_time','payment_type')

df_driver_date_summary = df_fares \
            .groupBy('driver_id','newDate') \
            .agg(
                F.sum('tip').alias('tip_total'),
                F.sum('tolls').alias('tolls_total'),
                F.sum('total_fare').alias('fares_total')
            ) \
            .withColumn('money_total', (col('fares_total')+col('tip_total')-col('tolls_total'))) \
            .drop('tip_total','tolls_total','fares_total') \

df_driver_date_summary.show(10)
```

```
+----------+----------+-----------------+
| driver_id|   newDate|      money_total|
+----------+----------+-----------------+
|2013000004|2013-01-01|634.9999964237213|
|2013000156|2013-01-01|397.4999998807907|
|2013000269|2013-01-01|388.0999993085861|
|2013000390|2013-01-01| 536.699998497963|
|2013000878|2013-01-01|            176.0|
|2013001608|2013-01-01|            186.0|
|2013001844|2013-01-01|601.1599986553192|
|2013001870|2013-01-01|312.7999997138977|
|2013002159|2013-01-01|574.2000073194504|
|2013002550|2013-01-01|587.3999952077866|
+----------+----------+-----------------+
only showing top 10 rows
```

Took 5 sec. Last updated by a.jourdan-dsti at March 17 2020, 4:04:26 PM.

**Compute by hour:**

FINISHED

- The average tip
- The average duration
- The average “distance at the crow flies” (“à vol d’oiseau”) - optional

Took 0 sec. Last updated by a.jourdan-dsti at March 17 2020, 9:50:24 PM.

FINISHED

```
%spark2.pyspark
from pyspark.sql import functions as F


df_all = complete_rides
df_all.show(10)

#z.show(taxi_fares)
```

```
+-------+----------+----------+-------------------+------------+---+-----+----------+-----------
--------+----------+---------+----------+---------+---------------+
|ride_id|   taxi_id| driver_id|         start_time|payment_type|tip|tolls|total_fare|
end_time| start_lon|start_lat|   end_lon|  end_lat|passenger_count|
+-------+----------+----------+-------------------+------------+---+-----+----------+-----------
--------+----------+---------+----------+---------+---------------+
|    148|2013000148|2013000148|2013-01-01 00:01:00|         CRD|5.6|  0.0|      34.1|2013-01-01
00:31:00| -73.95757|40.722225|  -73.9823|40.768288|              6|
|    463|2013000462|2013000460|2013-01-01 00:03:00|         CSH|0.0|  0.0|      12.0|2013-01-01
00:17:00| -73.99193|40.721977| -73.97819|40.745796|              1|
|    471|2013000470|2013000468|2013-01-01 00:03:00|         CSH|0.0|  0.0|      10.5|2013-01-01
00:10:00| -73.99046|40.731068| -73.96442| 40.75617|              2|
|    496|2013000495|2013000493|2013-01-01 00:03:14|         CSH|0.0|  0.0|       9.5|2013-01-01
00:11:04| -73.97381| 40.78436| -73.96638|40.764835|              1|
|    833|2013000137|2013000137|2013-01-01 00:05:00|         CSH|0.0|  0.0|       5.5|2013-01-01
00:08:00| -73.99266|40.737465|-73.992935|40.742664|              1|
|   1088|2013001075|2013001072|2013-01-01 00:06:00|         CSH|0.0|  0.0|      12.5|2013-01-01
00:17:00|-73.952965| 40.78325|-73.976524| 40.74413|              1|
```



Took 30 sec. Last updated by a.jourdan-dsti at March 17 2020, 9:51:19 PM.

FINISHED

```
%spark2.pyspark
from pyspark.sql import functions as F

df_summary_hour = df_all \
                    .withColumn('Hour',F.hour(df_all.start_time)) \
                    .withColumn('duration',df_all.end_time.cast('integer')-df_all.start_time.cas
                    .groupBy('Hour') \
                    .agg(
                        F.avg('tip').alias('avg_tip'),
                        F.avg('duration').alias('avg_dur')
                        ) \
                    .orderBy('Hour')

df_summary_hour.show(10)
```

```
+----+------------------+-----------------+
|Hour|           avg_tip|          avg_dur|
+----+------------------+-----------------+
|   0|1.1545570365368878|698.6189151833872|
|   1|1.1358809759312294|742.0401194545955|
|   2|1.1220456987138108|716.0527883899181|
|   3| 1.133566351323682|703.3400949376419|
|   4|1.2020738179234514|713.4099966479198|
|   5|  1.44166622713591| 727.837379274341|
|   6|1.1696094127827703|625.1184537090426|
|   7|1.1268225219193495|629.1714817657722|
|   8|1.1890459200905068|679.2368647935896|
|   9| 1.126201820113226|667.6123428090957|
+----+------------------+-----------------+
only showing top 10 rows
```

Took 27 sec. Last updated by a.jourdan-dsti at March 17 2020, 9:55:58 PM. (outdated)

FINISHED

### Find the hour of the day when the tips are the highest

Took 0 sec. Last updated by a.jourdan-dsti at March 17 2020, 9:55:49 PM.

```
%spark2.pyspark
```

FINISHED

```
df_hour_max_tip = df_summary_hour \
                    .orderBy(df_summary_hour.avg_tip.desc()) \
                    .limit(1)
```

dsti-a19/ajourdan/lab2

```
+----+---------------+---------------+
|Hour|        avg_tip|        avg_dur|
+----+---------------+---------------+
|   5|1.44166622713591|727.837379274341|
+----+---------------+---------------+
```

Took 30 sec. Last updated by a.jourdan-dsti at March 17 2020, 9:59:41 PM.

**Find the percentage of each type of payment** FINISHED

Took 0 sec. Last updated by a.jourdan-dsti at March 17 2020, 9:59:29 PM.

```
%spark2.pyspark
from pyspark.sql import functions as F

df_pay_percentage = df_all \
                    .groupBy('payment_type') \
                    .agg(
                        F.count('ride_id').alias('nb')
                        )

z.show(df_pay_percentage)
```

FINISHED

settings

| payment_type |
|---|
| CSH |
| DIS |
| CRD |
| UNK |
| NOC |

Took 13 sec. Last updated by a.jourdan-dsti at March 17 2020, 10:17:24 PM. (outdated)

**Find the top 10 drivers that:**
**Won the most money**
**Did the greater number of rides** FINISHED

Took 0 sec. Last updated by a.jourdan-dsti at March 17 2020, 10:06:04 PM.

FINISHED

```
%spark2.pyspark
from pyspark.sql import functions as F

df_driver_money_rides = df_all \
                    .groupBy('driver_id') \
                    .agg(
                        F.sum('total_fare').alias('tot_fare'),
                        F.count('ride_id').alias('nb_rides')
                        )

df_driver_money_top10 = df_driver_money_rides \
                    .orderBy(df_driver_money_rides.tot_fare.desc()) \
                    .limit(10)

df_driver_money_top10.show(15)
```

```
+----------+-----------------+--------+
| driver_id|         tot_fare|nb_rides|
+----------+-----------------+--------+
|2013009052|4002.5500135421753|     183|
|2013003398| 2866.199990749359|     209|
|2013010182| 2462.870002746582|      33|
|2013005950|2357.4700026512146|     152|
|2013008459|2302.4899954795837|     175|
|2013000801|2280.5499925613403|     163|
|2013003091|2245.5299944877625|     170|
|2013005290| 2242.610005378723|     135|
|2013006941|2241.7700033187866|     113|
|2013009947| 2235.069999217987|     128|
+----------+-----------------+--------+
```

Took 13 sec. Last updated by a.jourdan-dsti at March 17 2020, 10:15:56 PM. (outdated)

FINISHED

```
%spark2.pyspark
df_driver_rides_top10 = df_driver_money_rides \
                    .orderBy(df_driver_money_rides.nb_rides.desc()) \
                    .limit(10)

df_driver_rides_top10.show(15)
```

```
+----------+-----------------+--------+
| driver_id|         tot_fare|nb_rides|
+----------+-----------------+--------+
|2013000230|1433.4000067710876|     228|
|2013003398| 2866.199990749359|     209|
|2013007575| 2159.289999485016|     201|
|2013006255| 2148.979996204376|     186|
|2013009052|4002.5500135421753|     183|
|2013010227| 2145.730000972748|     177|
|2013008459|2302.4899954795837|     175|
|2013003091|2245.5299944877625|     170|
|2013002949|1719.6600017547607|     169|
|2013011744|2121.7500019073486|     165|
+----------+-----------------+--------+
```

Took 13 sec. Last updated by a.jourdan-dsti at March 17 2020, 10:16:12 PM.

READY

```
%spark2.pyspark
```

# dsti-a19/ajourdan/lab2