flights_example_DataFrame

October 5, 2020

[3]:

```
from pyspark import sql
from pyspark.sql import functions as f,udf


sqlContext = sql.SparkSession.builder \
    .master("local") \
    .appName("Flight DF") \
    .getOrCreate()

flights = sqlContext.read.format('csv')\
    .options(header='true', inferSchema='true')\
    .load("flights.csv.bz2")

airport = sqlContext.read.format('csv')\
    .options(header='true', inferSchema='true')\
    .load("airports.csv.bz2")
```

[43]:

```
linesHeader = lines.first()
flights_raw = lines\
    .zipWithIndex()\
    .filter(lambda x: x[1] > 2)\
    .keys()

flights = flights_raw\
    .map(lambda x: x.split(','))\
    .map(lambda x: (x[0], x[1], x[2], x[3], x[4],
                    x[5], x[6], x[7], x[8], x[9],
                    x[10], 0 if x[11]=='' else float(x[11]), x[12], x[13],␣
 ↪x[14],
                    x[15], x[16], x[17], x[18], x[19],
                    x[20], x[21], x[22], x[23], x[24],
                    x[25], x[26], x[27], x[28], x[29], x[30]
))


airports_raw = airports_lines\
    .zipWithIndex()\
    .filter(lambda x:x[1]>2)\
    .keys()
airports = airports_raw\
    .map(lambda x: x.split(','))
mainFlightsData = flights.map(lambda p:
                              (p[0], p[1], p[2], p[3], p[4], p[5], p[6],
                               p[7], p[8], p[9], p[10], p[11], p[24]))
```

```
      ␣
↪---------------------------------------------------------------------------

       NameError                                 Traceback (most recent call␣
↪last)

       <ipython-input-43-b883e54e11f3> in <module>
    ----> 1 linesHeader = lines.first()
          2 flights_raw = lines\
          3     .zipWithIndex()\
          4     .filter(lambda x: x[1] > 2)\
          5     .keys()


       NameError: name 'lines' is not defined
```

[4]:
```
flights.printSchema()
airport.printSchema()
```

```
root
 |-- YEAR: integer (nullable = true)
 |-- MONTH: integer (nullable = true)
 |-- DAY: integer (nullable = true)
 |-- DAY_OF_WEEK: integer (nullable = true)
 |-- AIRLINE: string (nullable = true)
 |-- FLIGHT_NUMBER: integer (nullable = true)
 |-- TAIL_NUMBER: string (nullable = true)
 |-- ORIGIN_AIRPORT: string (nullable = true)
 |-- DESTINATION_AIRPORT: string (nullable = true)
 |-- SCHEDULED_DEPARTURE: integer (nullable = true)
 |-- DEPARTURE_TIME: integer (nullable = true)
 |-- DEPARTURE_DELAY: integer (nullable = true)
 |-- TAXI_OUT: integer (nullable = true)
 |-- WHEELS_OFF: integer (nullable = true)
 |-- SCHEDULED_TIME: integer (nullable = true)
 |-- ELAPSED_TIME: integer (nullable = true)
 |-- AIR_TIME: integer (nullable = true)
 |-- DISTANCE: integer (nullable = true)
 |-- WHEELS_ON: integer (nullable = true)
 |-- TAXI_IN: integer (nullable = true)
 |-- SCHEDULED_ARRIVAL: integer (nullable = true)
 |-- ARRIVAL_TIME: integer (nullable = true)
 |-- ARRIVAL_DELAY: integer (nullable = true)
 |-- DIVERTED: integer (nullable = true)
 |-- CANCELLED: integer (nullable = true)
 |-- CANCELLATION_REASON: string (nullable = true)
 |-- AIR_SYSTEM_DELAY: integer (nullable = true)
 |-- SECURITY_DELAY: integer (nullable = true)
 |-- AIRLINE_DELAY: integer (nullable = true)
 |-- LATE_AIRCRAFT_DELAY: integer (nullable = true)
 |-- WEATHER_DELAY: integer (nullable = true)

root
 |-- IATA_CODE: string (nullable = true)
 |-- AIRPORT: string (nullable = true)
 |-- CITY: string (nullable = true)
 |-- STATE: string (nullable = true)
 |-- COUNTRY: string (nullable = true)
 |-- LATITUDE: double (nullable = true)
 |-- LONGITUDE: double (nullable = true)
```

[7]:

```
#Q1 Find a list of Origin Airports
flights.select("ORIGIN_AIRPORT").distinct().show()
```

```
+--------------+
|ORIGIN_AIRPORT|
+--------------+
|           BGM|
|           PSE|
|           INL|
|           DLG|
|         12888|
|           MSY|
|           PPG|
|         12003|
|         15041|
|           GEG|
|           SNA|
|           BUR|
|           GRB|
|           GTF|
|         14986|
|         13851|
|           IDA|
|          11150|
|          15412|
|            GRR|
+---------------+
only showing top 20 rows
```

[8]:

```
#Q2 Find a list of (Origin, Destination) pairs
flights.select("ORIGIN_AIRPORT", "DESTINATION_AIRPORT").distinct().show()
```

```
+--------------+-------------------+
|ORIGIN_AIRPORT|DESTINATION_AIRPORT|
+--------------+-------------------+
|           BQN|                MCO|
|           PHL|                MCO|
|           MCI|                IAH|
|           SPI|                ORD|
|           SNA|                PHX|
|           LBB|                DEN|
|           ORD|                PDX|
|           EWR|                STT|
|           ATL|                GSP|
|           MCI|                MKE|
|           PBI|                DCA|
|           SMF|                BUR|
|           MDW|                MEM|
|           LAS|                LIT|
|           TPA|                ACY|
|           DSM|                EWR|
|           FSD|                ATL|
|           SJC|                LIH|
|           CLE|                SJU|
|         11298|              11057|
+--------------+-------------------+
only showing top 20 rows
```

[10]:

```
#Q3 Find the Origin airport which had the largest departure delay in the month␣
 ↪of January
flights.where(flights.MONTH == 1)\
    .orderBy("DEPARTURE_DELAY", ascending=False)\
    .limit(1)\
    .select("ORIGIN_AIRPORT")\
    .show()
```

```
+--------------+
|ORIGIN_AIRPORT|
+--------------+
|           BHM|
+--------------+
```

[11]:

```
#Q4 Find out which carrier has the largest delay on Weekends.
flights.filter("DAY_OF_WEEK = 6 OR DAY_OF_WEEK = 7" )\
    .orderBy("DEPARTURE_DELAY", ascending=False)\
    .limit(1)\
    .select("AIRLINE")\
    .show()
```

```
+-------+
|AIRLINE|
+-------+
|     AA|
+-------+
```

[12]:

```
#Q5 Which airport has the most cancellation of flights?
flights.filter("CANCELLED = 1")\
    .withColumn("COUNT", f.lit(1))\
    .groupBy("ORIGIN_AIRPORT")\
    .agg(f.sum("COUNT").alias("COUNT"))\
    .orderBy("COUNT", ascending=False)\
    .limit(1)\
    .select("ORIGIN_AIRPORT", "COUNT")\
    .show()
```

```
+--------------+-----+
|ORIGIN_AIRPORT|COUNT|
+--------------+-----+
|           ORD| 8548|
+--------------+-----+
```

[14]:

```
#Q6 Find the percent of flights cancelled for each carrier.

flights.withColumn("TOTAL", f.lit(1))\
    .groupBy("AIRLINE")\
    .agg(f.sum("CANCELLED").alias("TOTAL_CANCELLED"), f.sum("TOTAL").
 ↪alias("TOTAL"))\
    .withColumn("CANCEL_RATE", f.col("TOTAL_CANCELLED")/f.col("TOTAL")*100)\
    .show()
```

```
+-------+---------------+------+------------------+
|AIRLINE|TOTAL_CANCELLED| TOTAL|       CANCEL_RATE|
+-------+---------------+------+------------------+
|     UA|           6573|515723| 1.274521400053905|
|     NK|           2004| 117379| 1.7072900604026275|
|     AA|          10919| 725984| 1.5040276369727157|
|     EV|          15231| 571977| 2.6628693111785964|
|     B6|           4276| 267048| 1.6012102693148795|
|     DL|           3824| 875881| 0.4365889886868193|
|     OO|           9960| 588353| 1.6928612584621818|
|     F9|            588|  90836| 0.6473204456382932|
|     US|           4067| 198715| 2.0466497244797823|
|     MQ|          15025| 294632| 5.0995818512585185|
|     HA|            171|  76272|0.22419760855884205|
|     AS|            669| 172521|   0.38777887909298|
|     VX|            534|  61903| 0.8626399366751207|
|     WN|          16043|1261855| 1.2713822111098343|
+-------+---------------+-------+-------------------+
```

[16]:

```
#Q7 Find the largest departure delay for each carrier

flights.groupBy("AIRLINE")\
    .agg(f.max("DEPARTURE_DELAY").alias("MAX_DEPARTURE_DELAY"))\
    .show()
```

```
+-------+-------------------+
|AIRLINE|MAX_DEPARTURE_DELAY|
+-------+-------------------+
|     UA|               1314|
|     NK|                836|
|     AA|               1988|
|     EV|               1274|
|     B6|               1006|
|     DL|               1289|
|     OO|               1378|
|     F9|               1112|
|     US|                759|
|     MQ|               1544|
|     HA|               1433|
|     AS|                963|
|     VX|                644|
|     WN|                665|
+-------+-------------------+
```

[17]:

```
#Q8 Find the largest departure delay for each carrier for each month
flights.groupBy("AIRLINE", "MONTH")\
    .agg(f.max("DEPARTURE_DELAY").alias("MAX_DEPARTURE_DELAY"))\
    .show()
```

```
+-------+-----+-------------------+
|AIRLINE|MONTH|MAX_DEPARTURE_DELAY|
+-------+-----+-------------------+
|     NK|   11|                476|
|     VX|   10|                430|
|     UA|   12|               1194|
|     HA|   10|               1022|
|     OO|    3|                874|
|     OO|    4|                878|
|     OO|    9|                893|
|     F9|    2|                852|
|     F9|   12|                781|
|     HA|    5|                326|
|     UA|    4|               1314|
|     MQ|   10|               1544|
|     HA|   12|               1095|
|     EV|    4|                757|
|     DL|    6|               1201|
|     DL|    3|               1166|
|     DL|    8|               1207|
|     B6|    6|                507|
|     DL|   10|               1120|
|     OO|   10|               1122|
+-------+-----+-------------------+
only showing top 20 rows
```

[19]:

```python
#Q9 For each carrier find the average Departure delay
flights.withColumn("TOTAL", f.lit(1))\
    .groupBy("AIRLINE")\
    .agg(f.sum("DEPARTURE_DELAY").alias("TOTAL_DEPARTURE_DELAY"), f.
 ↪sum("TOTAL").alias("TOTAL"))\
    .withColumn("AVG_DEPARTURE_DELAY", f.col("TOTAL_DEPARTURE_DELAY")/f.
 ↪col("TOTAL"))\
    .show()
```

```
+-------+---------------------+-------+-------------------+
|AIRLINE|TOTAL_DEPARTURE_DELAY|  TOTAL|AVG_DEPARTURE_DELAY|
+-------+---------------------+-------+-------------------+
|     UA|              7355348| 515723|  14.26220664969373|
|     NK|              1840887| 117379|  15.68327383944317|
|     AA|              6369435| 725984|  8.773519802089302|
|     EV|              4857338| 571977|    8.4921911195729|
|     B6|              3026467| 267048| 11.333044995656211|
|     DL|              6427294| 875881|  7.338090448359994|
|     OO|              4517510| 588353|   7.67823058605973|
|     F9|              1205449|  90836|  13.27060856928971|
|     US|              1196447| 198715| 6.0209194071912036|
|     MQ|              2837908| 294632|   9.63204268375465|
|     HA|               36972|  76272|0.48473882945248586|
|     AS|              306997| 172521| 1.7794761217474975|
|     VX|              553852|  61903|  8.947094648078446|
|     WN|            13186520|1261855| 10.450107183471951|
+-------+--------------------+-------+-------------------+
```

[20]:

```
#Q10 For each carrier find the average Departure delay for each month
flights.withColumn("TOTAL", f.lit(1))\
    .groupBy("AIRLINE","MONTH")\
    .agg(f.sum("DEPARTURE_DELAY").alias("TOTAL_DEPARTURE_DELAY"), f.
 ↪sum("TOTAL").alias("TOTAL"))\
    .withColumn("AVG_DEPARTURE_DELAY", f.col("TOTAL_DEPARTURE_DELAY")/f.
 ↪col("TOTAL"))\
    .select("AIRLINE", "MONTH", "AVG_DEPARTURE_DELAY")
    .show()
```

```
+-------+-----+---------------------+-----+-------------------+
|AIRLINE|MONTH|TOTAL_DEPARTURE_DELAY|TOTAL|AVG_DEPARTURE_DELAY|
+-------+-----+---------------------+-----+-------------------+
|     NK|   11|                87001|10164|  8.559720582447856|
|     VX|   10|                38540| 5464|  7.053440702781844|
|     UA|   12|               761043|43443|  17.51819625716456|
|     HA|   10|                 1049| 6242| 0.1680551105414931|
|     OO|    3|               289928|50078|  5.789528335796158|
|     OO|    4|               260302|49329|  5.276855399460763|
|     OO|    9|               182835|47625| 3.8390551181102364|
|     F9|    2|               146727| 5809| 25.258564296780857|
|     F9|   12|               129059| 8120|  15.89396551724138|
|     HA|    5|                -8676| 6434|-1.3484612993472178|
|     UA|    4|               532506|41342| 12.880508925547868|
|     MQ|   10|                75123|21982| 3.4174779364934946|
|     HA|   12|                -2771| 6260|-0.4426517571884984|
|     EV|    4|               328999|49296|  6.673949204803635|
|     DL|    6|               837824|77255| 10.844916186654585|
|     DL|    3|               622004|74166|  8.386646172100424|
|     DL|    8|               626586|80947|   7.74069452851866|
|     B6|    6|               255272|22558|  11.31625144073056|
|     DL|   10|               242914|75552| 3.2151895383312157|
|     OO|   10|               177362|48808| 3.6338714964759875|
+-------+-----+---------------------+-----+-------------------+
only showing top 20 rows
```

[22]:

```
#Q11 Which date of year has the highest rate  of flight cancellations?
# Rate of flight cancellation is calculated by deviding number of canceled␣
 ↪flights by total number of flights.
```

```
flights.withColumn("TOTAL", f.lit(1))\
    .groupBy("YEAR","MONTH","DAY")\
    .agg(f.sum("CANCELLED").alias("TOTAL_CANCELLED"), f.sum("TOTAL").
 ↪alias("TOTAL"))\
    .withColumn("CANCEL_RATE", f.col("TOTAL_CANCELLED")/f.col("TOTAL")*100)\
    .orderBy("CANCEL_RATE", ascending=False)\
    .limit(1)\
    .select("YEAR","MONTH","DAY")\
    .show()
```

```
+----+-----+---+
|YEAR|MONTH|DAY|
+----+-----+---+
|2015|    1| 27|
+----+-----+---+
```

[48]:

```
#Q12 Calculate the number of flights to each destination state
# For each carrier, for which state do they have the largest average delay?
# You will need the airline and airport data sets for this question.

from pyspark.sql.types import ArrayType, IntegerType, StringType
from pyspark.sql.functions import udf
fold_list = udf(lambda x,y: sorted(zip(x,y))[-1][1],StringType())

#Q8 Find the largest departure delay for each carrier for each month
flights.withColumn("COUNT", f.lit(1))\
    .groupBy("AIRLINE", "DESTINATION_AIRPORT")\
    .agg(f.sum("DEPARTURE_DELAY").alias("TOTAL_DEPARTURE_DELAY"),f.sum("COUNT").
 ↪alias("COUNT"))\
    .withColumn("DEPARTURE_AVG_DELAY", f.col("TOTAL_DEPARTURE_DELAY")/f.
 ↪col("COUNT"))\
    .join(airport, flights.DESTINATION_AIRPORT == airport.IATA_CODE)\
    .select("AIRLINE", "TOTAL_DEPARTURE_DELAY", "STATE")\
    .groupBy("AIRLINE")\
    .agg(
        f.collect_list("TOTAL_DEPARTURE_DELAY").alias("delay"),
        f.collect_list("STATE").alias("state")
    )\
    .withColumn("MAX_AVGDELAY_STATE", fold_list(f.col("delay"), f.
 ↪col("state")))\
    .select("AIRLINE", "MAX_AVGDELAY_STATE")\
    .show()
```

```
+-------+------------------+
|AIRLINE|MAX_AVGDELAY_STATE|
+-------+------------------+
|     UA|                IL|
|     NK|                IL|
|     AA|                TX|
|     EV|                GA|
|     B6|                NY|
|     DL|                GA|
|     OO|                IL|
|     F9|                CO|
|     US|                NC|
|     MQ|                IL|
|     HA|                HI|
|     AS|                WA|
|     VX|                CA|
|     WN|                IL|
+-------+------------------+
```

[ ]: