Spark Assignment 21.2 : Aviation Data Analysis

The assignment is about Aviation data analysis The dataset used for this assignment is given in the following links:
Delayed_Flights.csv

Delayed_Flights.csv Datasets

There are 29 columns in this dataset. Some of them have been mentioned below:

- Year: 1987 – 2008
- Month: 1 – 12
- FlightNum: Flight number
- Canceled: Was the flight cancelled?
- CancelleationCode: The reason for cancellation.

This assignment is done in the spark shell within the acadgildVM.

Problem Statement 1:

Find out the top 5 most visited destinations.

Steps Followed:

1) Copied the dataset file in the path /home/acadgild/spark/21_2.DelayedFlights.csv. Then read the text file by using sc.textfile as below.

```
val delayed_flights = sc.textFile("/home/acadgild/spark/21_2.DelayedFlights.csv")
```

```
scala> val delayed_flights = sc.textFile("/home/acadgild/spark/21_2.DelayedFlights.csv")
delayed_flights: org.apache.spark.rdd.RDD[String] = /home/acadgild/spark/21_2.DelayedFlights.csv MapPartitionsRDD[59] at textFile at
 <console>:24
```

2) Then calling the map function over delayed_flights, for each row, split it over ",", over the delayed_flights 's 18th element, which specifies origin, and add 1 as value for each origin. Then calling the reduceBy key, where the values for each key are aggregated using the given reduce function. Here we are adding the values for each key and finally using take function to display top 5 items.

```
val mapping = delayed_flights.map(x => x.split(",")).map(x => (x(18),1)).filter(x => x._1!=null).reduceByKey(_+_).map(x => (x._2,x._1)).sortByKey(false).map(x => (x._2,x._1)).take(5)
```

Output

mapping: Array[(String, Int)] = Array((ORD,108984), (ATL,106898), (DFW,70657), (DEN,63003), (LAX,59969))

Spark-shell output

```
scala> val delayed_flights = sc.textFile("/home/acadgild/spark/21_2.DelayedFlights.csv")
delayed_flights: org.apache.spark.rdd.RDD[String] = /home/acadgild/spark/21_2.DelayedFlights.csv MapPartitionsRDD[59] at textFile at
 <console>:24

scala> val mapping = delayed_flights.map(x => x.split(",")).map(x => (x(18),1)).filter(x => x._1!=null).reduceByKey(_+_).map(x => (x
._2,x._1)).sortByKey(false).map(x => (x._2,x._1)).take(5)
mapping: Array[(String, Int)] = Array((ORD,108984), (ATL,106898), (DFW,70657), (DEN,63003), (LAX,59969))
```

Problem Statement 2:

Which month has seen the most number of cancellations due to bad weather?

Steps Followed:

1) Copied the dataset file in the path /home/acadgild/spark/21_2.DelayedFlights.csv. Then read the text file by using sc.textfile as below.

```
val delayed_flights = sc.textFile("/home/acadgild/spark/21_2.DelayedFlights.csv")
```

```
scala> val delayed_flights = sc.textFile("/home/acadgild/spark/21_2.DelayedFlights.csv")
delayed_flights: org.apache.spark.rdd.RDD[String] = /home/acadgild/spark/21_2.DelayedFlights.csv MapPartitionsRDD[59] at textFile at
 <console>:24
```

2) Then calling the map function over delayed_flights, for each row, split it over ",", and filter the delayed_flights 's 22nd element, which specifies cancelled status "1" and 23rd element which is the cancellation code equals "B" for bad weather, keeping filtered item as key and add 1 as value for each origin.

   Then calling the reduceBy key, where the values for each key are aggregated using the given reduce function. Here we are adding the values for each key and finally using take function to display top 1 item.

```
val canceled = delayed_flights.map(x => x.split(",")).filter(x => ((x(22).equals("1"))&&(x(23).equals("B")))).map(x => (x(2),1)).reduceByKey(_+_).map(x => (x._2,x._1)).sortByKey(false).map(x => (x._2,x._1)).take(1)
```

Output

canceled: Array[(String, Int)] = Array((12,250))

Spark-shell output

```
scala> val canceled = delayed_flights.map(x => x.split(",")).filter(x => ((x(22).equals("1"))&&(x(23).equals("B")))).map(x => (x(2),
1)).reduceByKey(_+_).map(x => (x._2,x._1)).sortByKey(false).map(x => (x._2,x._1)).take(1)
canceled: Array[(String, Int)] = Array((12,250))
```

Problem Statement 3:

Top ten origins with the highest AVG departure delay

Steps Followed:

1) Copied the dataset file in the path /home/acadgild/spark/21_2.DelayedFlights.csv. Then read the text file by using sc.textfile as below.

   val delayed_flights = sc.textFile("/home/acadgild/spark/21_2.DelayedFlights.csv")

```
scala> val delayed_flights = sc.textFile("/home/acadgild/spark/21_2.DelayedFlights.csv")
delayed_flights: org.apache.spark.rdd.RDD[String] = /home/acadgild/spark/21_2.DelayedFlights.csv MapPartitionsRDD[59] at textFile at
 <console>:24
```

2) Then calling the map function over delayed_flights, for each row, split it over ",", over the delayed_flights 's 16th and 17th element, which specifies Departure delay and origin, then add 1 as value for each origin. Then calling the reduceBy key, where the values for each key are aggregated using the given reduce function. Here we are adding the values for each key. Then calculating the average departure delay and finally using take function to display top 10 items.

```
val avg = delayed_flights.map(x => x.split(",")).map(x => (x(17),x(16).toInt)).mapValues((_, 1)).reduceByKey((x, y) => (x._1 + y._1, x._2 + y._2)).mapValues{ case (sum, count) => (1.0 * sum)/count}.map(x => (x._2,x._1)).sortByKey(false).map(x => (x._2,x._1)).take(10)
```

Output

avg: Array[(String, Double)] = Array((CMX,154.95238095238096), (PLN,106.83333333333333), (SPI,86.05932203389831), (MOT,79.98571428571428), (ACY,79.3103448275862), (MQT,78.9776119402985), (HHH,75.55319148936171), (MBS,74.82413793103449), (ABI,74.80188679245283), (ACK,74.38461538461539))

Spark-shell output

```
scala> val avg = delayed_flights.map(x => x.split(",")).map(x => (x(17),x(16).toInt)).mapValues((_, 1)).reduceByKey((x, y) => (x._1
+ y._1, x._2 + y._2)).mapValues{ case (sum, count) => (1.0 * sum)/count}.map(x => (x._2,x._1)).sortByKey(false).map(x => (x._2,x._1)
).take(10)
avg: Array[(String, Double)] = Array((CMX,154.95238095238096), (PLN,106.83333333333333), (SPI,86.05932203389831), (MOT,79.9857142857
1428), (ACY,79.3103448275862), (MQT,78.9776119402985), (HHH,75.55319148936171), (MBS,74.82413793103449), (ABI,74.80188679245283), (A
CK,74.38461538461539))
```

Problem Statement 4:

Which route (origin & destination) has seen the maximum diversion?

Steps Followed:

1) Copied the dataset file in the path /home/acadgild/spark/21_2.DelayedFlights.csv. Then read the text file by using sc.textfile as below.

   val delayed_flights = sc.textFile("/home/acadgild/spark/21_2.DelayedFlights.csv")

```
scala> val delayed_flights = sc.textFile("/home/acadgild/spark/21_2.DelayedFlights.csv")
delayed_flights: org.apache.spark.rdd.RDD[String] = /home/acadgild/spark/21_2.DelayedFlights.csv MapPartitionsRDD[59] at textFile at
 <console>:24
```

2) Then calling the map function over delayed_flights, for each row, split it over ",", and filter the delayed_flights 's 24th element, which specifies CarrierDelay equals "1" and using map function, keeping origin and destination as key and adding 1 to value.

   Then calling the reduceBy key, where the values for each key are aggregated using the given reduce function. Here we are adding the values for each key and finally using take function to display top 10 item.

```
val diversion = delayed_flights.map(x => x.split(",")).filter(x => ((x(24).equals("1")))).map(x => ((x(17)+","+x(18)),1)).reduceByKey(_+_).map(x => (x._2,x._1)).sortByKey(false).map(x => (x._2,x._1)).take(10).foreach(println)
```

Output

(ORD,LGA,39)
(DAL,HOU,35)
(DFW,LGA,33)
(ATL,LGA,32)
(SLC,SUN,31)
(ORD,SNA,31)
(MIA,LGA,31)
(BUR,JFK,29)
(HRL,HOU,28)
(BUR,DFW,25)

Spark-shell output

```
scala> val diversion = delayed_flights.map(x => x.split(",")).filter(x => ((x(24).equals("1")))).map(x => ((x(17)+","+x(18)),1)).red
uceByKey(_+_).map(x => (x._2,x._1)).sortByKey(false).map(x => (x._2,x._1)).take(10).foreach(println)
(ORD,LGA,39)
(DAL,HOU,35)
(DFW,LGA,33)
(ATL,LGA,32)
(SLC,SUN,31)
(ORD,SNA,31)
(MIA,LGA,31)
(BUR,JFK,29)
(HRL,HOU,28)
(BUR,DFW,25)
diversion: Unit = ()
```