

Assignment 28

The U.S. Department of Transportation's (DOT) Bureau of Transportation Statistics (BTS) tracks the on-time performance of domestic flights operated by large air carriers.

Summary information on the number of on-time, delayed, canceled, and diverted flights appears in DOT's monthly Air Travel Consumer Report, published about 30 days after the month's end, as well as in summary tables posted on this website. Summary statistics and raw data are made available to the public at the time the Air Travel Consumer Report is released.

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 canceled?
- CancellationCode: The reason for cancellation.

Problem Statement 1

Find out the top 5 most visited destinations.

Step1: Load the dataset from `/user/acadgild/DelayedFlights.csv` and put to RDD `delayed_flights`. Code is as below:

```
val delayed_flights = sc.textFile("/user/acadgild/DelayedFlights.csv")
```

 acadgild@localhost:~

```
scala> val delayed_flights = sc.textFile("/user/acadgild/DelayedFlights.csv")
delayed_flights: org.apache.spark.rdd.RDD[String] = /user/acadgild/DelayedFlight
s.csv MapPartitionsRDD[1] at textFile at <console>:24
scala> █
```

Step2: Get the header

```
val header = delayed_flights.first()
```

```
scala> val header = delayed_flights.first()
header: String = ,Year,Month,DayOfMonth,DayOfWeek,DepTime,CRSDepTime,ArrTime,CRSArrTime,UniqueCarrier,FlightNum,TailNum,ActualElapsedTime,CRSElapsedTime,AirTime,ArrDelay,DepDelay,Origin,Dest,Distance,TaxiIn,TaxiOut,Canceled,CancellationCode,Diverted,CarrierDelay,WeatherDelay,NASDelay,SecurityDelay,LateAircraftDelay
scala>
```

Step3: Remove header from delayed_flights and put to delayed_flights_no_header

val delayed_flights_no_header = delayed_flights.filter(x=> x != header)

```
scala> val delayed_flights_no_header = delayed_flights.filter(x=> x != header)
delayed_flights_no_header: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[2] at filter at <console>:28
scala>
```

Step4: Query to Find out the top 5 most visited destinations

- Split the fields of delayed_flights_no_header based on separator comma.
- Map the Dest field (which is field 18) to 1
- Filter only Not Null records
- Use reduceByKey to get total number of time Dest appears group by Dest
- Sort the list by number of time Dest appears
- Take the first 5
- Print the tuples

```
val mapping = delayed_flights_no_header.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).foreach(println)
```

```
acadgild@localhost:~  
scala> val mapping = delayed_flights_no_header.map(x => x.split(",")).map(x => (x(10),1)).filter(x => x._1!=null).reduceByKey(_+_).map(x => (x._2,x._1)).sortByKey(false)  
) .map(x => (x._2,x._1)).take(5).foreach(println)  
(ORD,108994)  
(ATL,106898)  
(DFW,70657)  
(DEN,63003)  
(LAX,59969)  
mapping: Unit = ()  
  
scala> █
```

Problem Statement 2

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

Step1: Load the dataset from /user/acadgild/DelayedFlights.csv and put to RDD delayed_flights. Code is as below:

```
val delayed_flights = sc.textFile("/user/acadgild/DelayedFlights.csv")
```

```
acadgild@localhost:~  
scala> val delayed_flights = sc.textFile("/user/acadgild/DelayedFlights.csv")  
delayed_flights: org.apache.spark.rdd.RDD[String] = /user/acadgild/DelayedFlight  
s.csv MapPartitionsRDD[1] at textFile at <console>:24  
  
scala> █
```

Step2: Get the header

```
val header = delayed_flights.first()
```

```
scala> val header = delayed_flights.first()
header: String = ,Year,Month,DayOfMonth,DayOfWeek,DepTime,CRSDepTime,ArrTime,CRSArrTime,UniqueCarrier,FlightNum,TailNum,ActualElapsedTime,CRSElapsedTime,AirTime,ArrDelay,DepDelay,Origin,Dest,Distance,TaxiIn,TaxiOut,Canceled,CancellationCode,Diverted,CarrierDelay,WeatherDelay,NASDelay,SecurityDelay,LateAircraftDelay
scala>
```

Step3: Remove header from delayed_flights and put to delayed_flights_no_header

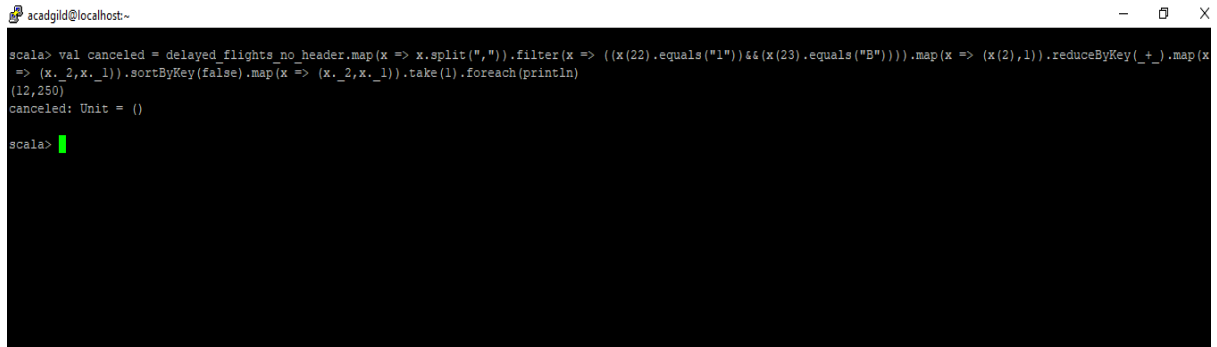
val delayed_flights_no_header = delayed_flights.filter(x=> x != header)

```
scala> val delayed_flights_no_header = delayed_flights.filter(x=> x != header)
delayed_flights_no_header: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[2] at filter at <console>:28
scala>
```

Step4: Query the delayed_flights to get Which month has seen the most number of cancellations due to bad weather.

- Split the fields of delayed_flights_no_header based on separator comma.
- Map the Dest field (which is field 18) to 1
- Filter those records for which Canceled field (field 22) is 1
- Filter those records for which Cancellation field (field 230) is B
- Use map to get Month , count as 1
- Use reduceByKey to get total number of time Month appears group by Month
- Sort the list by number of time Month appears
- Take the first 1
- Print the tuple

```
val canceled = delayed_flights_no_header.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).foreach(println)
```



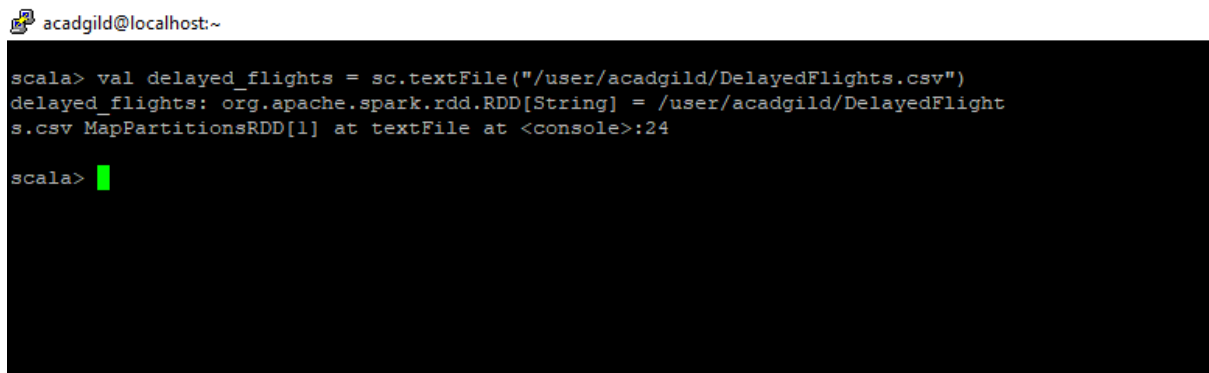
```
acadgild@localhost:~
scala> val canceled = delayed_flights_no_header.map(x => x.split(",")).filter(x => ((x(22).equals("1"))&&(x(23).equals("B")))).map(x => (x(2),1)).reduceByKey(_+_).map(x
=> (x._2,x._1)).sortByKey(false).map(x => (x._2,x._1)).take(1).foreach(println)
(12,250)
canceled: Unit = ()
scala>
```

Problem Statement 3

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

Step1: Load the dataset from /user/acadgild/DelayedFlights.csv and put to RDD delayed_flights. Code is as below:

```
val delayed_flights = sc.textFile("/user/acadgild/DelayedFlights.csv")
```



```
acadgild@localhost:~
scala> val delayed_flights = sc.textFile("/user/acadgild/DelayedFlights.csv")
delayed_flights: org.apache.spark.rdd.RDD[String] = /user/acadgild/DelayedFlight
s.csv MapPartitionsRDD[1] at textFile at <console>:24
scala>
```

Step2: Get the header

```
val header = delayed_flights.first()
```

```
scala> val header = delayed_flights.first()
header: String = ,Year,Month,DayOfMonth,DayOfWeek,DepTime,CRSDepTime,ArrTime,CRSArrTime,UniqueCarrier,FlightNum,TailNum,ActualElapsedTime,CRSElapsedTime,AirTime,ArrDelay,DepDelay,Origin,Dest,Distance,TaxiIn,TaxiOut,Cancelled,CancellationCode,Diverted,CarrierDelay,WeatherDelay,NASDelay,SecurityDelay,LateAircraftDelay
scala>
```

Step3: Remove header from delayed_flights and put to delayed_flights_no_header

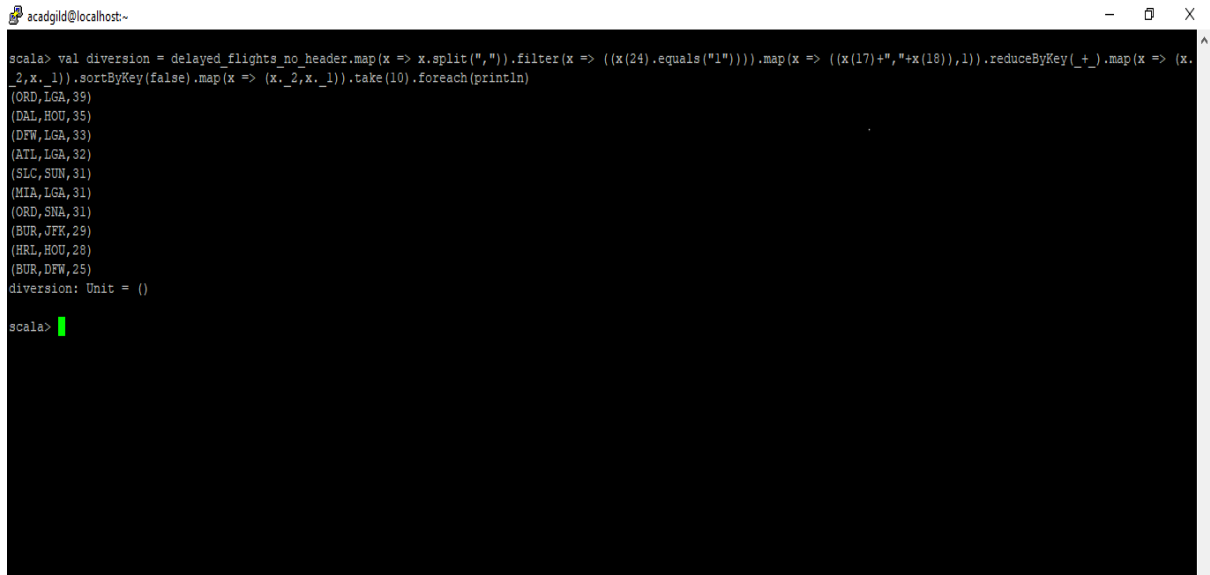
val delayed_flights_no_header = delayed_flights.filter(x=> x != header)

```
scala> val delayed_flights_no_header = delayed_flights.filter(x=> x != header)
delayed_flights_no_header: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[2] at filter at <console>:28
scala>
```

Step4: Query to get Which route (origin & destination) has seen the maximum diversion

- Split the fields of delayed_flights_no_header based on separator comma.
- Filter tuples for which Diverted field is 1
- Using map, concatenate Origin, Dest to 1
- Use reduceByKey to get total number of diversions by Source, Destination
- Sort by total number of diversions descending
- Take first 10
- Print the Origin, Destination Number of Diversions

```
val diversion = delayed_flights_no_header.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)
```



```
scala> val diversion = delayed_flights_no_header.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)
(ORD,LGA,39)
(DAL,HOU,35)
(DFW,LGA,33)
(ATL,LGA,32)
(SLC,SUN,31)
(MIA,LGA,31)
(ORD,SNA,31)
(BUR,JFK,29)
(HRL,HOU,28)
(BUR,DFW,25)
diversion: Unit = ()

scala>
```