**Assignment 21.2:**

Implement the below blog at your end and send the complete documentation.

<https://drive.google.com/file/d/0B_Qjau8wv1KoUThzZ24tT1NsZGs/view?usp=sharing>

Aviation data analysis:

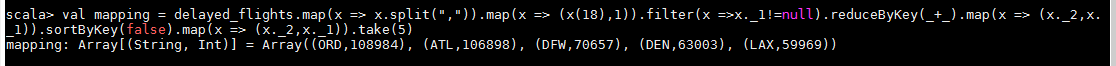
**Problem Statement 1**

Find out the top 5 most visited destinations.

val delayed\_flights =sc.textFile("DelayedFlights.csv")

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)



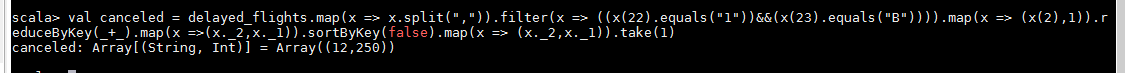


**Problem Statement 2**

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

val delayed\_flights = sc.textFile("DelayedFlights.csv")

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)



**Problem Statement 3**

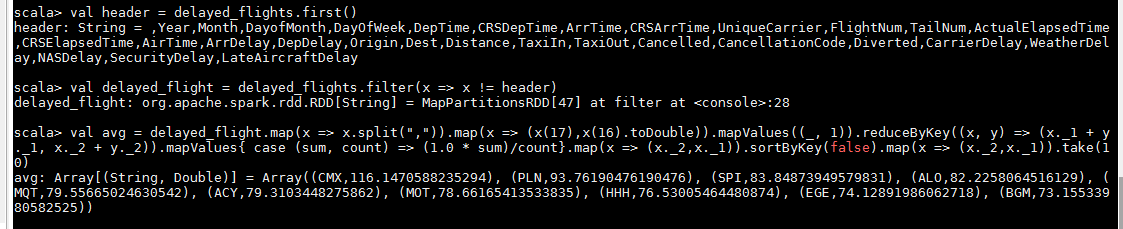
Top ten origins with the highest AVG departure delay.

val header = delayed\_flights.first()

val delayed\_flight = delayed\_flights.filter(x => x != header)

val avg = delayed\_flight.map(x => x.split(",")).map(x => (x(17),x(16).toDouble)).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)





**Problem Statement 4:**

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

val delayed\_flights = sc.textFile("DelayedFlights.csv")

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)

