Sentiment analysis on demonetization

Here in this assignment we are trying to find out the views of different people on the demonetization by analyzing the tweets from twitter.

We have two input datasets-

1. demonetization-tweets.csv- It contains the twitter tweets.
2. AFINN.txt- It contains the keywords for sentiment analysis.

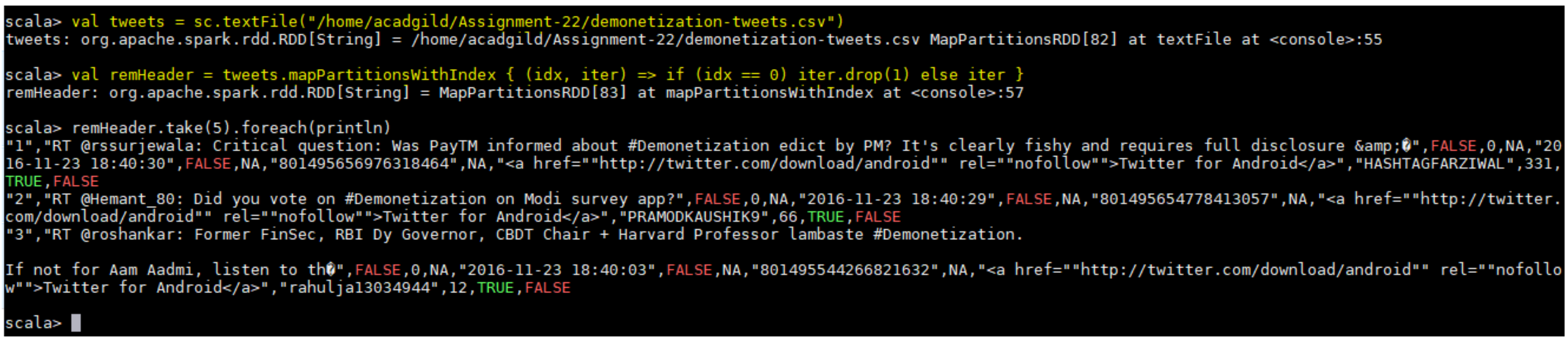
Below is the code used to achieve the required output-

- val tweets = sc.textFile("/home/acadgild/Assignment-22/demonetization-tweets.csv")
- val remHeader = tweets.mapPartitionsWithIndex { (idx, iter) => if (idx == 0) iter.drop(1) else iter }
- val tweetMap = remHeader.map(x => x.split(",")).filter(x=>x.length>=2).map(x => (x(0).replaceAll("\"",""),x(1).replaceAll("\"","").toLowerCase)).map(x => (x._1,x._2.split(" "))).toDF("id","words")
- tweetMap.registerTempTable("tweets")
- val explode = spark.sql("select id as id,explode(words) as word from tweets").registerTempTable("tweet_word")
- val afinn = sc.textFile("/home/acadgild/Assignment-22/AFINN.txt").map(x => x.split("\t")).map(x => (x(0),x(1))).toDF("word","rating").registerTempTable("afinn")
- val join = spark.sql("select t.id,AVG(a.rating) as rating from tweet_word t join afinn a on t.word=a.word group by t.id order by rating desc").show

Now we will try to understand each and every line of code one by one.

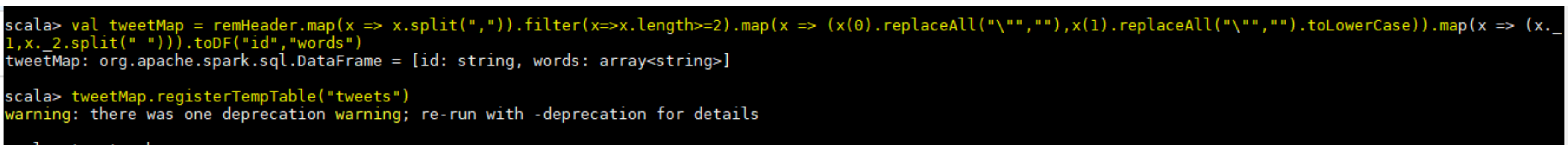
First we are reading the demonetization-tweets.csv file. Since this file contains header so we in next step we are removing the header from it.

Same can be seen in below screenshot-

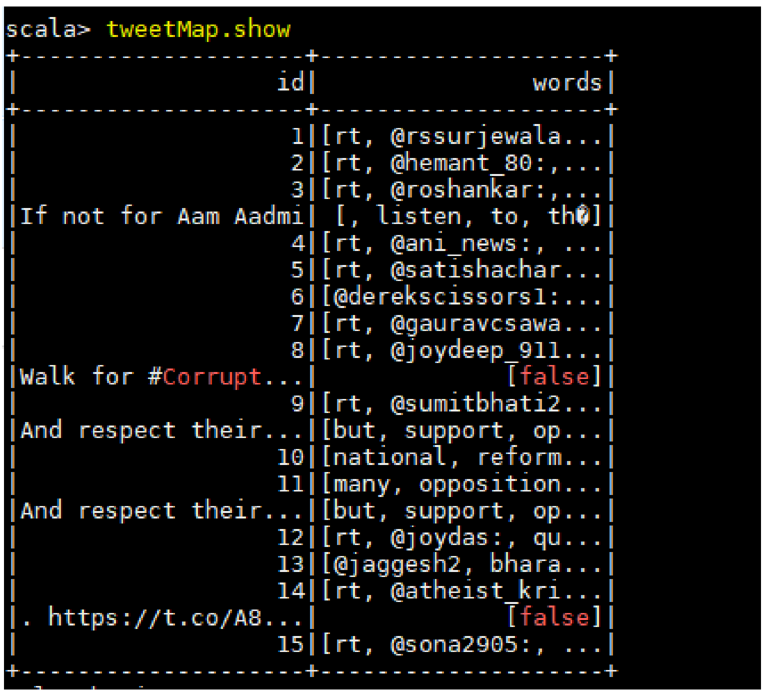


```
scala> val tweets = sc.textFile("/home/acadgild/Assignment-22/demonetization-tweets.csv")
tweets: org.apache.spark.rdd.RDD[String] = /home/acadgild/Assignment-22/demonetization-tweets.csv MapPartitionsRDD[82] at textFile at <console>:55

scala> val remHeader = tweets.mapPartitionsWithIndex { (idx, iter) => if (idx == 0) iter.drop(1) else iter }
remHeader: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[83] at mapPartitionsWithIndex at <console>:57

scala> remHeader.take(5).foreach(println)
"1","RT @rssurjewala: Critical question: Was PayTM informed about #Demonetization edict by PM? It's clearly fishy and requires full disclosure &amp;�",FALSE,0,NA,"20
16-11-23 18:40:30",FALSE,NA,"801495656976318464",NA,"<a href=""http://twitter.com/download/android"" rel=""nofollow"">Twitter for Android</a>","HASHTAGFARZIWAL",331,
TRUE,FALSE
"2","RT @Hemant_80: Did you vote on #Demonetization on Modi survey app?",FALSE,0,NA,"2016-11-23 18:40:29",FALSE,NA,"801495654778413057",NA,"<a href=""http://twitter.
com/download/android"" rel=""nofollow"">Twitter for Android</a>","PRAMODKAUSHIK9",66,TRUE,FALSE
"3","RT @roshankar: Former FinSec, RBI Dy Governor, CBDT Chair + Harvard Professor lambaste #Demonetization.

If not for Aam Aadmi, listen to th�",FALSE,0,NA,"2016-11-23 18:40:03",FALSE,NA,"801495544266821632",NA,"<a href=""http://twitter.com/download/android"" rel=""nofollo
w"">Twitter for Android</a>","rahulja13034944",12,TRUE,FALSE

scala> 
```

Now we are creating a dataframe from the above rdd which will contains some id and their corresponding words



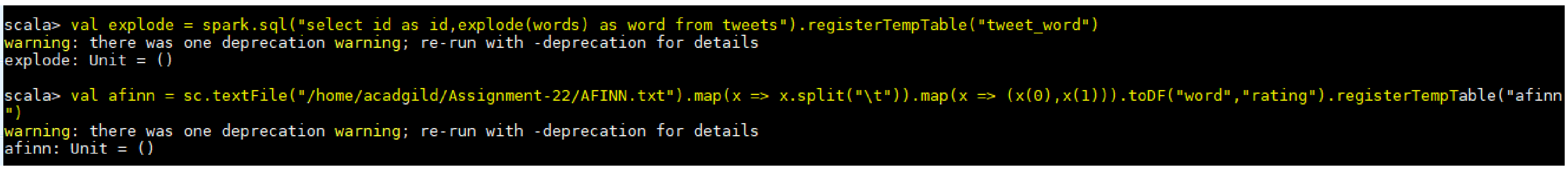
```
scala> val tweetMap = remHeader.map(x => x.split(",")).filter(x=>x.length>=2).map(x => (x(0).replaceAll("\"",""),x(1).replaceAll("\"","").toLowerCase)).map(x => (x._
1,x._2.split(" "))).toDF("id","words")
tweetMap: org.apache.spark.sql.DataFrame = [id: string, words: array<string>]

scala> tweetMap.registerTempTable("tweets")
warning: there was one deprecation warning; re-run with -deprecation for details
```



```
scala> tweetMap.show
+--------------------+--------------------+
|                  id|               words|
+--------------------+--------------------+
|                   1|[rt, @rssurjewala...|
|                   2|[rt, @hemant_80:,...|
|                   3|[rt, @roshankar:,...|
|If not for Aam Aadmi| [, listen, to, th�]|
|                   4|[rt, @ani_news:, ...|
|                   5|[rt, @satishachar...|
|                   6|[@derekscissors1:...|
|                   7|[rt, @gauravcsawa...|
|                   8|[rt, @joydeep_911...|
|Walk for #Corrupt...|             [false]|
|                   9|[rt, @sumitbhati2...|
|And respect their...|[but, support, op...|
|                  10|[national, reform...|
|                  11|[many, opposition...|
|And respect their...|[but, support, op...|
|                  12|[rt, @joydas:, qu...|
|                  13|[@jaggesh2, bhara...|
|                  14|[rt, @atheist_kri...|
|. https://t.co/A8...|             [false]|
|                  15|[rt, @sona2905:, ...|
+--------------------+--------------------+
```

Now we are using explode to create a temporary table named as “tweet_word”.

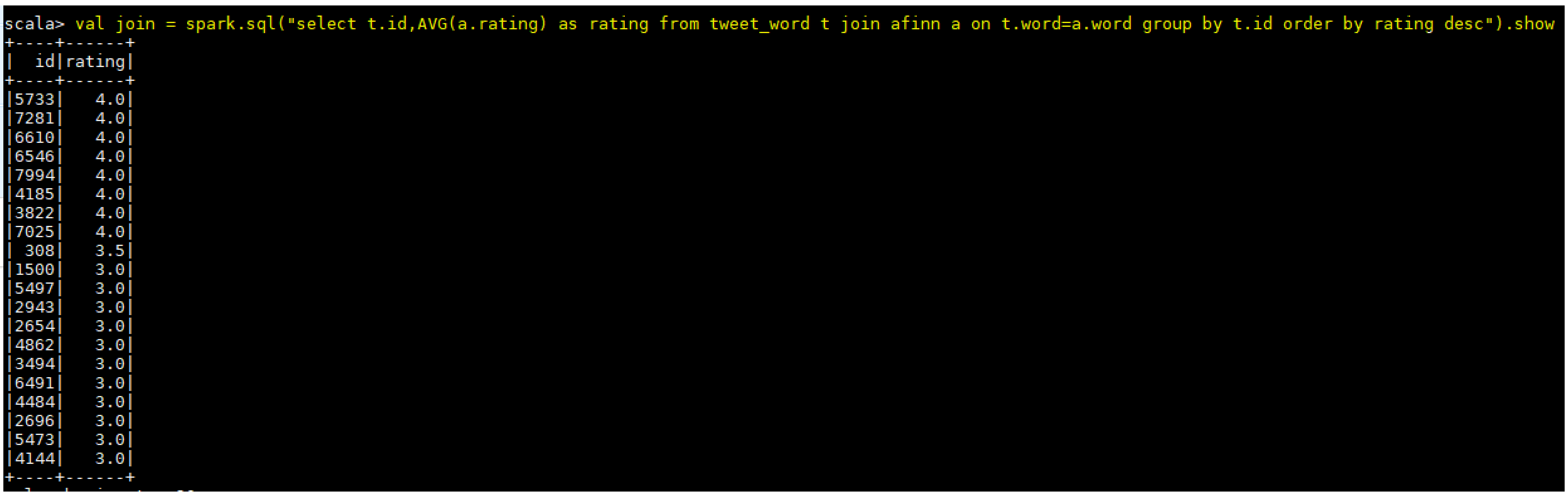
Then we are reading another file AFINN.txt and creating a temporary table with columns word and rating



```
scala> val explode = spark.sql("select id as id,explode(words) as word from tweets").registerTempTable("tweet_word")
warning: there was one deprecation warning; re-run with -deprecation for details
explode: Unit = ()

scala> val afinn = sc.textFile("/home/acadgild/Assignment-22/AFINN.txt").map(x => x.split("\t")).map(x => (x(0),x(1))).toDF("word","rating").registerTempTable("afinn
")
warning: there was one deprecation warning; re-run with -deprecation for details
afinn: Unit = ()
```

Finally we are joining both temporary tables “tweet_word” and “afinn” using the column word to find the id and rating.

Below screenshot shows the final output-



```
scala> val join = spark.sql("select t.id,AVG(a.rating) as rating from tweet_word t join afinn a on t.word=a.word group by t.id order by rating desc").show
+----+------+
|  id|rating|
+----+------+
|5733|   4.0|
|7281|   4.0|
|6610|   4.0|
|6546|   4.0|
|7994|   4.0|
|4185|   4.0|
|3822|   4.0|
|7025|   4.0|
| 308|   3.5|
|1500|   3.0|
|5497|   3.0|
|2943|   3.0|
|2654|   3.0|
|4862|   3.0|
|3494|   3.0|
|6491|   3.0|
|4484|   3.0|
|2696|   3.0|
|5473|   3.0|
|4144|   3.0|
+----+------+
```