PRINCIPLES OF BIG DATA MANAGEMENT

PHASE #1

TEAM SIZE: 4

TEAM MEMBERS: Sampath Gattu (sgvhb@mail.umkc.edu)
Moulika Chadalavada (mc7d8@mail.umkc.edu)
Prathyusha (ppf43@mail.umkc.edu)
Lakshmi Kona Nikitha (lnkwd5@mail.umkc.edu)

Objective:

- The main aim of this phase is to develop a system to store, analyze, and visualize a social network's.
- Tasks:
  1. Collect social network's data (e.g. tweets) in JSON format.
  2. Store the text content (e.g. tweet's text) from the data into a file in HDFS.
  3. Run a Word Count program in Apache Spark on the text file and store the output and log files locally.

Applications/Software's Used: Apache Spark, Hadoop, Scala, Cloudera, Twitter Developer Account, Python.

Collecting tweets from Twitter:

- Firstly, we have created a developer account in Twitter using below link.
  https://apps.twitter.com/

- Below are the variables that contains the user credentials to access Twitter API
  - ACCESS_TOKEN = "1974127951-N1QRXNCsVCywXl67BU1Wx7VlJ1fw2TlScZDY07s"
  - ACCESS_SECRET = "zLLfZ4ZF9BxvgzjXAkjJFAVFOL4P1i0fLSaZzc6LrnqZJ"
  - CONSUMER_KEY = "RfqrQLUtEAvwqSmNpGjZydmOn"
  - CONSUMER_SECRET = "1XdUvxMDfcL5eE89oAo7ZO8UESv6H7HacNa9B0Y7cGFa5MiPjo"

- We have written python program that is used to fetch tweets in JSON format. (Tweets-Collect.py)

- The extracted file in JSON format contains all the tweet details such as id, created at, text, profile_background_image_url etc.

- From JSON tweets file only the ***text*** content is extracted using Python program. The fetched text details are stored in a file. (TwitterTextConvert.py)

Store the text content (e.g. tweet's text) from the data into a file in HDFS.

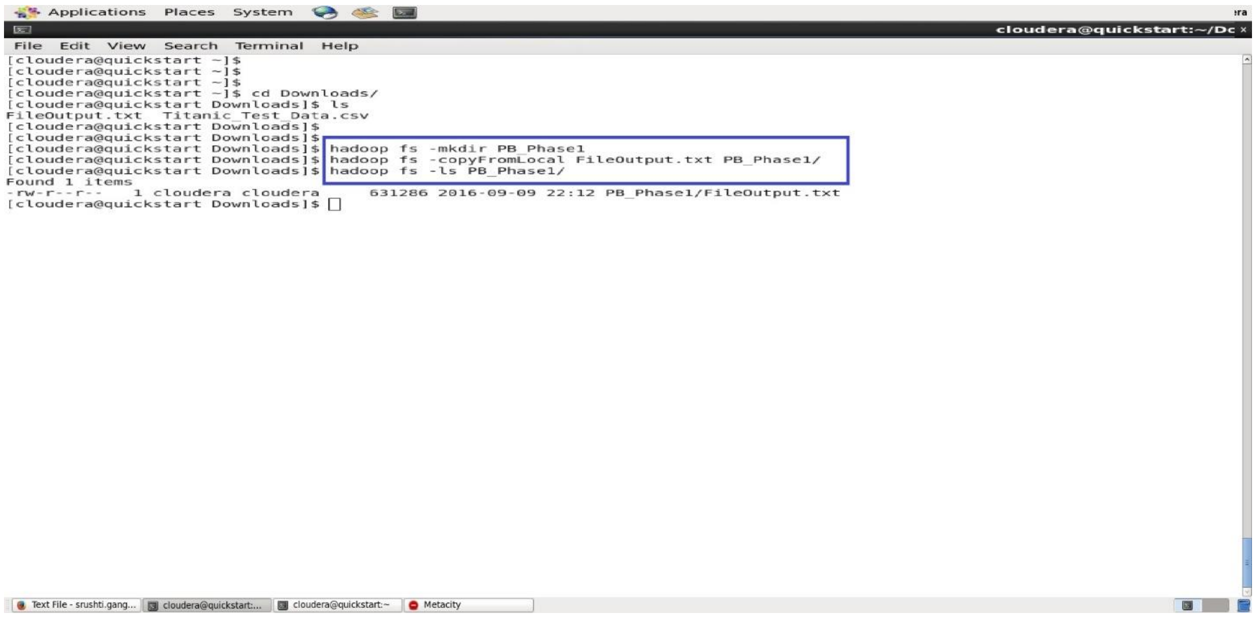
- The twitter tweets text content file is moved from local to HFDS.

- First a folder is created in HDFS and the text file is moved from local to HDFS using below command.

Create directory in local: hadoop fs -mkdir PB_Phase1

Move text file from local to HDFS: hadoop fs -copyFromLocal FileOutput.txt PB_Phase1/

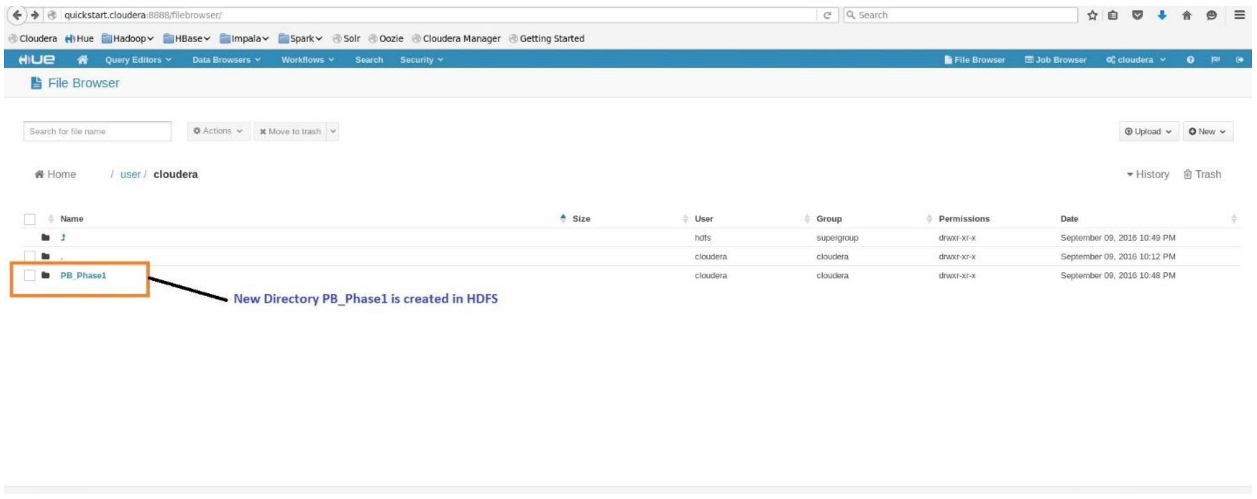
To list the files under a directory: hadoop fs -ls PB_Phase/



```
[cloudera@quickstart ~]$
[cloudera@quickstart ~]$
[cloudera@quickstart ~]$
[cloudera@quickstart ~]$ cd Downloads/
[cloudera@quickstart Downloads]$ ls
FileOutput.txt  Titanic_Test_Data.csv
[cloudera@quickstart Downloads]$
[cloudera@quickstart Downloads]$
[cloudera@quickstart Downloads]$ hadoop fs -mkdir PB_Phase1
[cloudera@quickstart Downloads]$ hadoop fs -copyFromLocal FileOutput.txt PB_Phase1/
[cloudera@quickstart Downloads]$ hadoop fs -ls PB_Phase1/
Found 1 items
-rw-r--r--   1 cloudera cloudera     631286 2016-09-09 22:12 PB_Phase1/FileOutput.txt
[cloudera@quickstart Downloads]$ 
```

Fig 1: HDFS Commands

- The directory created and the files moved to HDFS can be viewed as shown below.



quickstart.cloudera:8888/filebrowser/

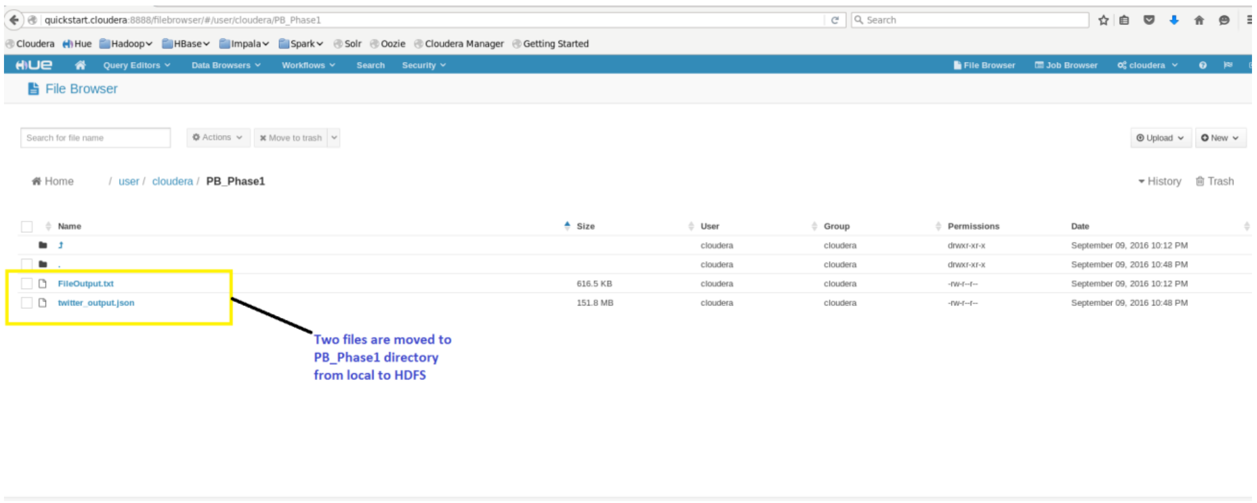
File Browser

Home / user / cloudera

Name	Size	User	Group	Permissions	Date
↥		hdfs	supergroup	drwxr-xr-x	September 09, 2016 10:49 PM
.		cloudera	cloudera	drwxr-xr-x	September 09, 2016 10:12 PM
PB_Phase1		cloudera	cloudera	drwxr-xr-x	September 09, 2016 10:48 PM

New Directory PB_Phase1 is created in HDFS

Fig 2: Directory in HDFS



quickstart.cloudera:8888/filebrowser/#/user/cloudera/PB_Phase1

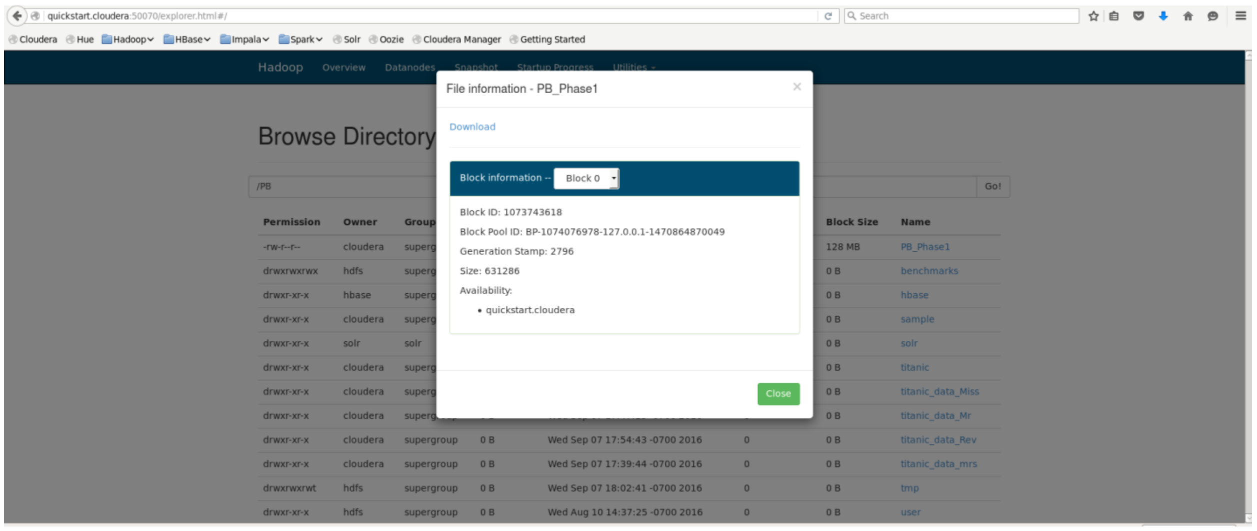
File Browser

Home / user / cloudera / PB_Phase1

Name	Size	User	Group	Permissions	Date
↥		cloudera	cloudera	drwxr-xr-x	September 09, 2016 10:12 PM
.		cloudera	cloudera	drwxr-xr-x	September 09, 2016 10:48 PM
FileOutput.txt	616.5 KB	cloudera	cloudera	-rw-r--r--	September 09, 2016 10:12 PM
twitter_output.json	151.8 MB	cloudera	cloudera	-rw-r--r--	September 09, 2016 10:48 PM

Two files are moved to PB_Phase1 directory from local to HDFS

Fig 3: Files in HDFS



quickstart.cloudera:50070/explorer.html#/

File information - PB_Phase1

Block information -- Block 0

Block ID: 1073743618

Block Pool ID: BP-1074076978-127.0.0.1-1470864870049

Generation Stamp: 2796

Size: 631286

Availability:

- quickstart.cloudera

Browse Directory

/PB

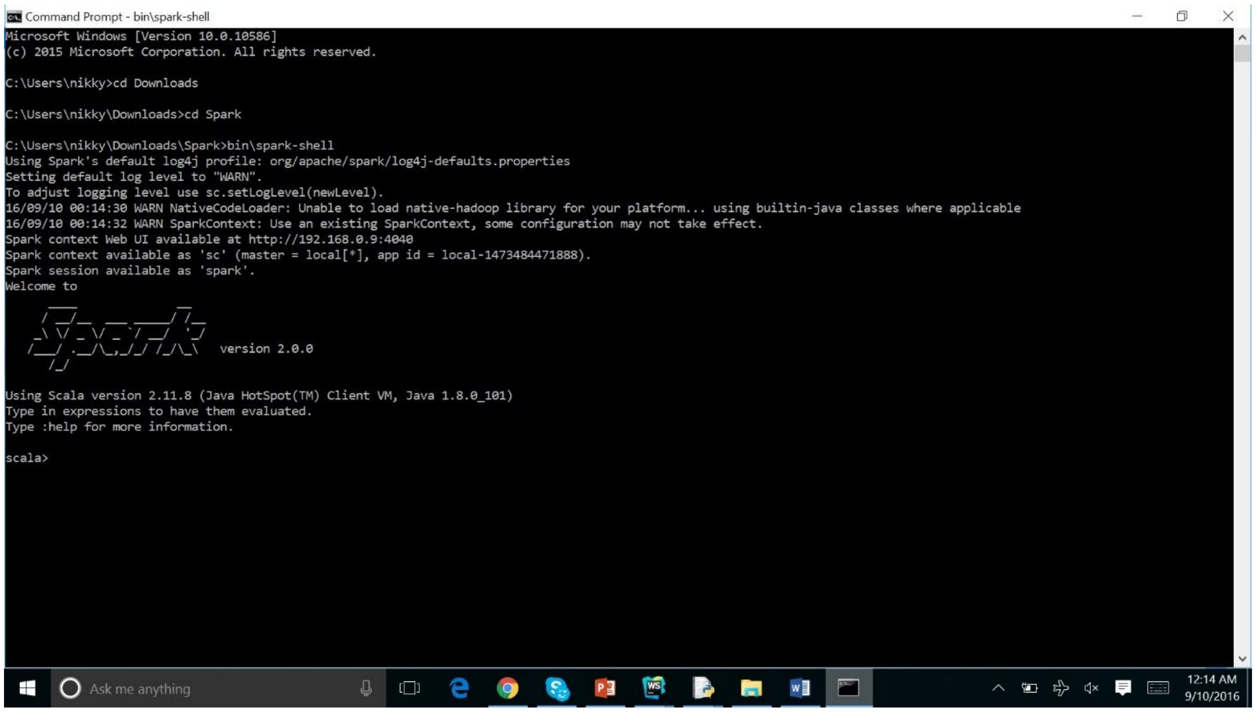
Permission	Owner	Group				Block Size	Name
-rw-r--r--	cloudera	superg				128 MB	PB_Phase1
drwxrwxrwx	hdfs	superg				0 B	benchmarks
drwxr-xr-x	hbase	superg				0 B	hbase
drwxr-xr-x	cloudera	superg				0 B	sample
drwxr-xr-x	solr	solr				0 B	solr
drwxr-xr-x	cloudera	superg				0 B	titanic
drwxr-xr-x	cloudera	superg				0 B	titanic_data_Miss
drwxr-xr-x	cloudera	superg				0 B	titanic_data_Mr
drwxr-xr-x	cloudera	supergroup	0 B	Wed Sep 07 17:54:43 -0700 2016	0	0 B	titanic_data_Rev
drwxr-xr-x	cloudera	supergroup	0 B	Wed Sep 07 17:39:44 -0700 2016	0	0 B	titanic_data_mrs
drwxrwxrwt	hdfs	supergroup	0 B	Wed Sep 07 18:02:41 -0700 2016	0	0 B	tmp
drwxr-xr-x	hdfs	supergroup	0 B	Wed Aug 10 14:37:25 -0700 2016	0	0 B	user

Fig 4: Directory Information

Run a Word Count program in Apache Spark on the text file and store the output and log files locally.

- First of all, to run word count program on set of data we require Apache Spark and scala.
- Once the Spark and Scala is installed successfully, in cmd prompt once the spark directory with ***spark-shell*** is given the Spark is integrated on Scala.

    Eg: C:/Users/Nik/Desktop/Spark>bin\spark-shell

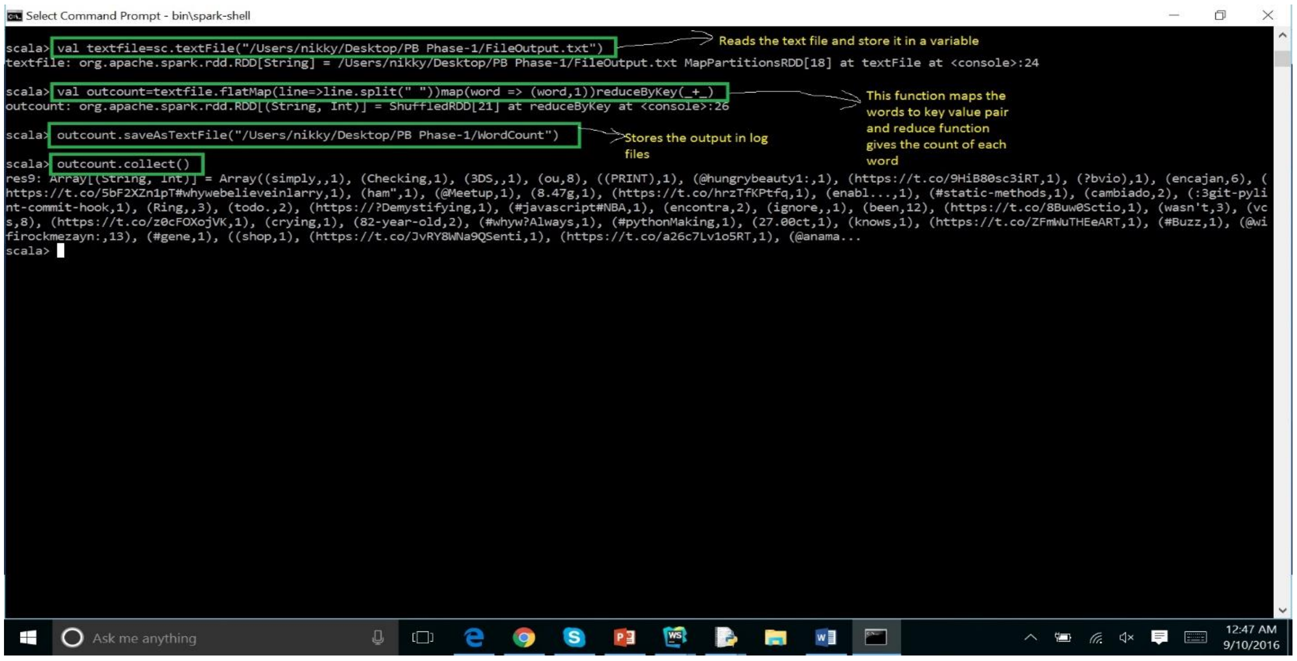


Command Prompt - bin\spark-shell

```
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\nikky>cd Downloads

C:\Users\nikky\Downloads>cd Spark

C:\Users\nikky\Downloads\Spark>bin\spark-shell
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel).
16/09/10 00:14:30 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
16/09/10 00:14:32 WARN SparkContext: Use an existing SparkContext, some configuration may not take effect.
Spark context Web UI available at http://192.168.0.9:4040
Spark context available as 'sc' (master = local[*], app id = local-1473484471888).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 2.0.0
      /_/

Using Scala version 2.11.8 (Java HotSpot(TM) Client VM, Java 1.8.0_101)
Type in expressions to have them evaluated.
Type :help for more information.

scala>
```

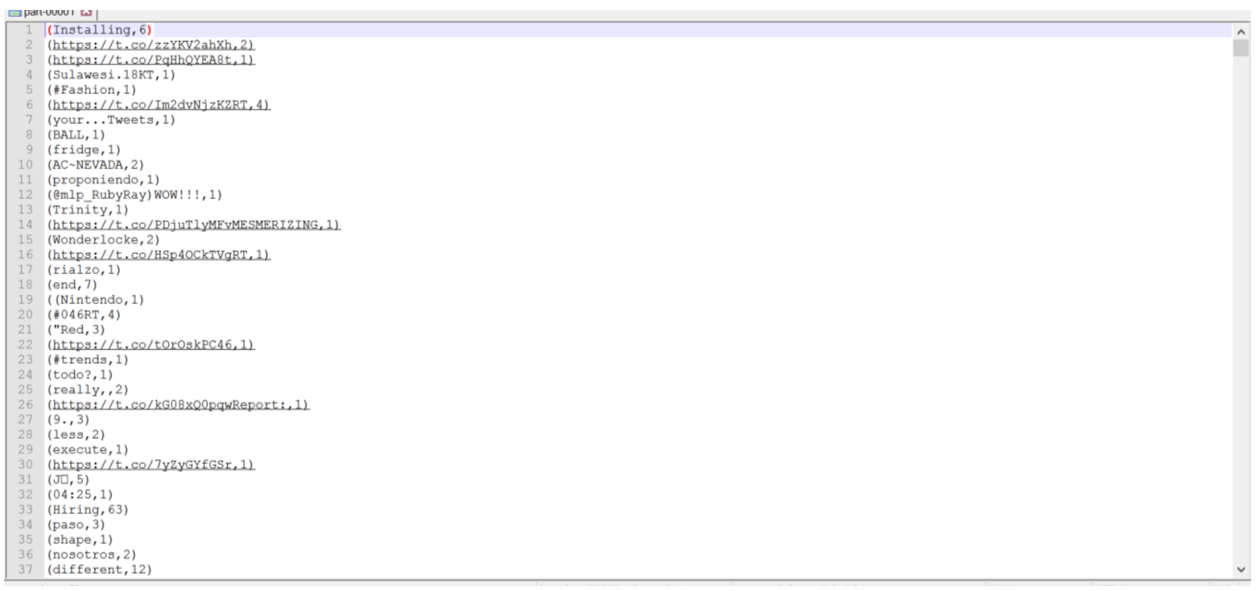
Fig 5: Spark

- Once spark is opened successfully then word count program is written and executed in ***scala.***
  1. **val textfile = sc.textFile("/Users/nikky/Desktop/PB Phase-1/FileOutput.txt")**
     - sc.textFile in Spark Shell, creates a RDD with each line as an element. For example, if there are 10 files in folder, 10 partitions will be created.
     - RDD means ***Resilient Distributed Datasets*** which are immutable and partitioned collection of records, which can only be created by coarse grained operations such as map, filter, group by etc.
     - RDDs can only be created by reading data from a stable storage such as HDFS or by transformations on existing RDDs.
  2. **val outcount = textFile.flatMap(line => line.split(" "))map(word => (word, 1))reduceByKey(_ + _)**
     - Once the file is read and stored in a variable textfile MapReduce function is used to execute word count program
     - MapReduce works by breaking the processing into 2 phases ***Map Phase and Reduce Phase.***
     - Each phase has key-value pairs as input and output, by specifying two functions ***Map Function and Reduce Function.***
  3. **outcount.saveAsTextFile("/Users/nikky/Desktop/PB Phase-1/WordCount")**
     - Once the reduce function group the words with respective count, the data is stored in WordCount folder.


Select Command Prompt - bin\spark-shell
Reads the text file and store it in a variable
This function maps the words to key value pair and reduce function gives the count of each word
Stores the output in log files
Ask me anything
12:47 AM
9/10/2016

```
scala> val textfile=sc.textFile("/Users/nikky/Desktop/PB Phase-1/FileOutput.txt")
textfile: org.apache.spark.rdd.RDD[String] = /Users/nikky/Desktop/PB Phase-1/FileOutput.txt MapPartitionsRDD[18] at textFile at <console>:24

scala> val outcount=textfile.flatMap(line=>line.split(" "))map(word => (word,1))reduceByKey(_+_)
outcount: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[21] at reduceByKey at <console>:26

scala> outcount.saveAsTextFile("/Users/nikky/Desktop/PB Phase-1/WordCount")

scala> outcount.collect()
res9: Array[(String, Int)] = Array((simply,,1), (Checking,1), (3DS,,1), (ou,8), ((PRINT),1), (@hungrybeauty1:,1), (https://t.co/9HiB80sc3iRT,1), (?bvio),1), (encajan,6), (
https://t.co/5bF2XZn1pT#whywebelieveinlarry,1), (ham",1), (@Meetup,1), (8.47g,1), (https://t.co/hrzTfKPtfq,1), (enabl...,1), (#static-methods,1), (cambiado,2), (:3git-pyli
nt-commit-hook,1), (Ring,,3), (todo.,2), (https://?Demystifying,1), (#javascript#NBA,1), (encontra,2), (ignore,,1), (been,12), (https://t.co/8Buw0Sctio,1), (wasn't,3), (vc
s,8), (https://t.co/z0cFOXojVK,1), (crying,1), (82-year-old,2), (#whyw?Always,1), (#pythonMaking,1), (27.00ct,1), (knows,1), (https://t.co/ZFmWuTHEeART,1), (#Buzz,1), (@wi
firockmezayn:,13), (#gene,1), ((shop,1), (https://t.co/JvRY8WNa9QSenti,1), (https://t.co/a26c7Lv1o5RT,1), (@anama...
scala> 
```

Fig 6: Word Count Commands



part-00001

```
(Installing,6)
(https://t.co/zzYKV2ahXh,2)
(https://t.co/PqHhQYEA8t,1)
(Sulawesi.18KT,1)
(#Fashion,1)
(https://t.co/Im2dvNjzKZRT,4)
(your...Tweets,1)
(BALL,1)
(fridge,1)
(AC~NEVADA,2)
(proponiendo,1)
(@mlp_RubyRay)WOW!!!,1)
(Trinity,1)
(https://t.co/PDjuTlyMFvMESMERIZING,1)
(Wonderlocke,2)
(https://t.co/HSp4OCkTVgRT,1)
(rialzo,1)
(end,7)
((Nintendo,1)
(#046RT,4)
("Red,3)
(https://t.co/tOrOskPC46,1)
(#trends,1)
(todo?,1)
(really,,2)
(https://t.co/kG08xQ0pqwReport:,1)
(9.,3)
(less,2)
(execute,1)
(https://t.co/7yZyGYfGSr,1)
(J□,5)
(04:25,1)
(Hiring,63)
(paso,3)
(shape,1)
(nosotros,2)
(different,12)
```

Fig 7: Sample Word Count Output