ULLAAS M
1NT19IS179
C2

4.

```
package ullaas;


import java.io.IOException;
import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
public class transactioncount {

//MAPPER CODE

public static class Map extends MapReduceBase implements
Mapper<LongWritable, Text, Text, IntWritable> {
private final static IntWritable one = new IntWritable(1);
//private Text word = new Text();

public void map(LongWritable key, Text value, OutputCollector<Text,
IntWritable> output, Reporter reporter) throws IOException {
String myString = value.toString();
String[] userCount = myString.split(",");
output.collect(new Text(userCount[3]), one);

}
}

//REDUCER CODE
public static class Reduce extends MapReduceBase implements
Reducer<Text, IntWritable, Text, IntWritable> {



public void reduce(Text key, Iterator<IntWritable> values,
OutputCollector<Text, IntWritable> output, Reporter reporter) throws
IOException { //{little: {1,1}}
int finaluserCount = 0 ;
Text mykey = key ;
while(values.hasNext()) {
IntWritable value = values.next();
finaluserCount += value.get();
}
output.collect(mykey, new IntWritable(finaluserCount));
}
}
//DRIVER CODE
public static void main(String[] args) throws Exception {
JobConf conf = new JobConf(transactioncount.class);
conf.setJobName("wordcount");
conf.setOutputKeyClass(Text.class);
conf.setOutputValueClass(IntWritable.class);
conf.setMapperClass(Map.class);
conf.setCombinerClass(Reduce.class);
conf.setReducerClass(Reduce.class);
conf.setInputFormat(TextInputFormat.class);
conf.setOutputFormat(TextOutputFormat.class); // hadoop jar
```

//jarname classpath inputfolder outputfolder
FileInputFormat.*setInputPaths*(conf, new Path(args[0]));
FileOutputFormat.*setOutputPath*(conf, new Path(args[1]));
JobClient.*runJob*(conf);
}

transactioncount.java

```java
package ullaas;


import java.io.IOException;
import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
public class transactioncount {

//MAPPER CODE

public static class Map extends MapReduceBase implements
Mapper<LongWritable, Text, Text, IntWritable> {
private final static IntWritable one = new IntWritable(1);
//private Text word = new Text();

public void map(LongWritable key, Text value, OutputCollector<Text,
IntWritable> output, Reporter reporter) throws IOException {
String myString = value.toString();
String[] userCount = myString.split(",");
output.collect(new Text(userCount[3]), one);

}
}

```

```java
//REDUCER CODE
public static class Reduce extends MapReduceBase implements
Reducer<Text, IntWritable, Text, IntWritable> {



public void reduce(Text key, Iterator<IntWritable> values,
OutputCollector<Text, IntWritable> output, Reporter reporter) throws
IOException { //{little: {1,1}}
int finaluserCount = 0 ;
Text mykey = key ;
while(values.hasNext()) {
IntWritable value = values.next();
finaluserCount += value.get();
}
output.collect(mykey, new IntWritable(finaluserCount));
}
}
//DRIVER CODE
public static void main(String[] args) throws Exception {
JobConf conf = new JobConf(transactioncount.class);
conf.setJobName("wordcount");
conf.setOutputKeyClass(Text.class);
conf.setOutputValueClass(IntWritable.class);
conf.setMapperClass(Map.class);
conf.setCombinerClass(Reduce.class);
conf.setReducerClass(Reduce.class);
```

```
//DRIVER CODE
public static void main(String[] args) throws Exception {
JobConf conf = new JobConf(transactioncount.class);
conf.setJobName("wordcount");
conf.setOutputKeyClass(Text.class);
conf.setOutputValueClass(IntWritable.class);
conf.setMapperClass(Map.class);
conf.setCombinerClass(Reduce.class);
conf.setReducerClass(Reduce.class);
conf.setInputFormat(TextInputFormat.class);
conf.setOutputFormat(TextOutputFormat.class); // hadoop jar
//jarname classpath inputfolder outputfolder
FileInputFormat.setInputPaths(conf, new Path(args[0]));
FileOutputFormat.setOutputPath(conf, new Path(args[1]));
JobClient.runJob(conf);
}
}
```

creating a csv file

	A	B	C	D
1	sl	card name	username	amount
2	1	Q	A	10000
3	2	W	B	2000
4	3	E	C	10000
5	4	R	D	65000

cd $HADOOP_HOME/sbin
./start-all.sh
jps

```
hdoop admin1-HP-280-G4-MT-Business-PC ~ cd $HADOOP_HOME/sbin

hdoop admin1-HP-280-G4-MT-Business-PC ~/hadoop-3.2.1/sbin ./start-all.sh
WARNING: Attempting to start all Apache Hadoop daemons as hdoop in 10 seconds.
WARNING: This is not a recommended production deployment configuration.
WARNING: Use CTRL-C to abort.
jps
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [admin1-HP-280-G4-MT-Business-PC]
Starting resourcemanager
Starting nodemanagers

hdoop admin1-HP-280-G4-MT-Business-PC ~/hadoop-3.2.1/sbin jps
9250 DataNode
9526 SecondaryNameNode
8134 org.eclipse.equinox.launcher_1.5.600.v20191014-2022.jar
9895 NodeManager
9096 NameNode
10330 Jps
9724 ResourceManager

hdoop admin1-HP-280-G4-MT-Business-PC ~/hadoop-3.2.1/sbin hdfs dfs -mkdir -p /input
```

hdfs dfs -mkdir -p /input

hdfs dfs -copyFromLocal /home/hdoop/Desktop/ULLAAS.csv /input

hadoop jar /home/hdoop/Desktop/ullaas.jar /input /output

hdfs dfs -cat /output/part*