LARGE SCALE DATA PROCESSING (CSE 3025)
WIN SEMESTER 2017-18

NAME: B.SHUBANKAR

REGNO: 15BCE1123

PROFESSOR: DR. Maheswari N

LAB-6

Customers details(customer name, address,mobile) are maintained in a file in hdfs. Create the customer details file with 10 records and do the following:

1. Use keyvalueinput format as the input format. Consider customer name as key. Separate the customer name and address,mobile using “,” separator in the text file. (Ex: ram,233 first street, chennai,9999967891 (or) ram,233 first street chennai 9999967891). Customer name is the key and Address, Mobile are the value.

PROGRAM:

```
import java.io.IOException;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.conf.*;

import org.apache.hadoop.io.*;

import org.apache.hadoop.mapreduce.*;

import org.apache.hadoop.mapreduce.lib.input.*;
```

```
import org.apache.hadoop.mapreduce.lib.output.*;

public class l6_1 {

public static class Map extends Mapper<Text,Text,Text,Text>

{

        String c="ram";

public void map(Text key, Text value, Context context) throws IOException,
InterruptedException

{

        String line=key.toString();

if(c.equalsIgnoreCase(line))

context.write(key,value);

}

}

public static void main(String[] args) throws Exception {

Configuration conf = new Configuration();

conf.set("mapreduce.input.keyvaluelinerecordreader.key.value.separator",",");

Job job = new Job(conf, "Customer");

job.setJarByClass(l6_1.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(Text.class);
```

job.setMapperClass(Map.class);

job.setInputFormatClass(KeyValueTextInputFormat.class);

FileInputFormat.addInputPath(job, new Path(args[0]));

FileOutputFormat.setOutputPath(job,new Path(args[1]));

job.waitForCompletion(true);

}

}

2.Write the customer details in mapper with 3 lines in each mapper. Use NLineinput format.

Note:

Use only mapper.

Mapper:

Context.write(key,value).

Driver:

conf.setInt(NLineInputFormat.LINES_PER_MAP, 3);

job.setInputFormatClass(NLineInputFormat.class);

job.setNumReduceTasks(0);

PROGRAM:

```
import java.io.IOException;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.conf.*;

import org.apache.hadoop.io.*;

import org.apache.hadoop.mapreduce.*;

import org.apache.hadoop.mapreduce.lib.input.*;

import org.apache.hadoop.mapreduce.lib.output.*;

publicclass l6_2 {

publicstaticclass Map extends Mapper<LongWritable,Text,Text,Text>

{

publicvoid map(Text key, Text value, Context context) throws IOException,
InterruptedException

{

        context.write(key, value);

}

}

publicstaticvoid main(String[] args) throws Exception {

Configuration conf = new Configuration();

// conf.set("mapreduce.input.keyvaluelinerecordreader.key.value.separator",",");
```

```
conf.setInt(NLineInputFormat.LINES_PER_MAP, 3);
Job job = new Job(conf, "Customer");
job.setJarByClass(l6_1.class);
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(Text.class);
job.setMapperClass(Map.class);
job.setInputFormatClass(NLineInputFormat.class);
job.setNumReduceTasks(0);
FileInputFormat.addInputPath(job, new Path(args[0]));
FileOutputFormat.setOutputPath(job,new Path(args[1]));
job.waitForCompletion(true);
}
}
```

3. Write the customer details using sequential file output and read using sequential file input format.

Note:

To write file:

Mapper:

Context.write(key,value);

Driver:

job.setOutputFormatClass(SequenceFileOutputFormat.class);

To read file:

Mapper:

Context.write(key,value);

Driver:

job.setInputFormatClass(SequenceFileInputFormat.class);

PROGRAM:

To write:

```
import java.io.IOException;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.conf.*;

import org.apache.hadoop.io.*;

import org.apache.hadoop.mapreduce.*;

import org.apache.hadoop.mapreduce.lib.input.*;

import org.apache.hadoop.mapreduce.lib.output.*;

publicclass l6_3 {

 publicstaticclass Map extends Mapper<LongWritable,Text,LongWritable,Text>

{

publicvoid map(LongWritable key, Text value, Context context) throws
IOException, InterruptedException
```

```
{
        context.write(key, value);
}
}
publicstaticvoid main(String[] args) throws Exception {
Configuration conf = new Configuration();
Job job = new Job(conf, "Customer");
job.setNumReduceTasks(0);
job.setJarByClass(l6_3.class);
job.setOutputKeyClass(LongWritable.class);
job.setOutputValueClass(Text.class);
job.setMapperClass(Map.class);
//job.setInputFormatClass(SequenceFileInputFormat.class);
job.setOutputFormatClass(SequenceFileOutputFormat.class);
FileInputFormat.addInputPath(job, new Path(args[0]));
FileOutputFormat.setOutputPath(job,new Path(args[1]));
job.waitForCompletion(true);
}
}
```

To read:

```
import java.io.IOException;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.conf.*;

import org.apache.hadoop.io.*;

import org.apache.hadoop.mapreduce.*;

import org.apache.hadoop.mapreduce.lib.input.*;

import org.apache.hadoop.mapreduce.lib.output.*;

 publicclass l6_3 {

 publicstaticclass Map extends Mapper<LongWritable,Text,LongWritable,Text>

{

publicvoid map(LongWritable key, Text value, Context context) throws
IOException, InterruptedException

{

        context.write(key, value);

}

}

publicstaticvoid main(String[] args) throws Exception {

Configuration conf = new Configuration();

Job job = new Job(conf, "Customer");
```

```
job.setNumReduceTasks(0);

job.setJarByClass(l6_3.class);

job.setOutputKeyClass(LongWritable.class);

job.setOutputValueClass(Text.class);

job.setMapperClass(Map.class);

job.setInputFormatClass(SequenceFileInputFormat.class);

//job.setOutputFormatClass(SequenceFileOutputFormat.class);

FileInputFormat.addInputPath(job, new Path(args[0]));

FileOutputFormat.setOutputPath(job,new Path(args[1]));

job.waitForCompletion(true);

}

}
```

OUTPUT: