Amjad Hamidi
202110401

Using HDFS commands , show how to do the following • create the input directory; name it ‘sportRetailer’ • the sample data above, add it to a text file name it ‘salesData.txt’, put this file in a directory called ‘input’ in the sportsRetailer on HDFS • browse the content of the salesData.txt on HDFS

```
Hadoop  fs -mkdir  sportRetailer

Hadoop  fs  -mkdir  sportRetailer/input

Hadoop  fs  -mkdir  sportRetailer/output4

Mkdir input

Cd input

Hadoop  fs   -put  salesData.txt   sportRetailer/input
```

Using HDFS commands, show to do the following: • run the job • browse the content of ‘output’ directory • show the content of the output files

```
Cd Desktop/

Hadoop  jar  SalesProject.jar  SalesRetailer   sportRetailer/input   sportRetailer/output4

Hadoop   fs  -ls   sportRetailer/output4

Hadoop  fs   -cat  sportRetailer/output4/part-r-*

Hadoop   fs  -cat  sportRetailer/output4/part-r-*  |  wc  -l
```

Full code => MapReduce Job

```
؛import java.io.DataInput

؛import java.io.DataOutput

؛import java.io.IOException


؛import org.apache.hadoop.conf.Configuration

؛import org.apache.hadoop.fs.Path

؛import org.apache.hadoop.io.DoubleWritable

؛import org.apache.hadoop.io.Text

؛import org.apache.hadoop.io.WritableComparable

؛import org.apache.hadoop.io.WritableComparator

؛import org.apache.hadoop.io.WritableUtils

؛import org.apache.hadoop.mapreduce.Job

؛import org.apache.hadoop.mapreduce.Mapper

؛import org.apache.hadoop.mapreduce.Mapper.Context

؛import org.apache.hadoop.mapreduce.Reducer

؛import org.apache.hadoop.mapreduce.lib.input.FileInputFormat

؛import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat


Made by : Amjad Hamidi 202110401 //

} <public class SalesRetailer implements WritableComparable<SalesRetailer

؛private Text salesRetailer

؛private DoubleWritable salesAmount


} ()public SalesRetailer

؛()this.salesRetailer = new Text
```

```
;()this.salesAmount = new DoubleWritable

{

} (public SalesRetailer(Text salesRetailer, DoubleWritable salesAmount

;this.salesRetailer = salesRetailer

;this.salesAmount = salesAmount

{

} ()public Text getSalesRetailer

;return salesRetailer

{

} (public void setSalesRetailer(Text salesRetailer

;this.salesRetailer = salesRetailer

{

} ()public DoubleWritable getSalesAmount

;return salesAmount

{

} (public void setSalesAmount(DoubleWritable salesAmount

;this.salesAmount = salesAmount

{

Override@

} public void write(DataOutput out) throws IOException

;(salesRetailer.write(out

;(salesAmount.write(out
{

Override@
} public void readFields(DataInput in) throws IOException
؛(salesRetailer.readFields(in
؛(salesAmount.readFields(in
{

Override@
} (public int compareTo(SalesRetailer o
؛(int cmp = -1 * this.salesAmount.compareTo(o.salesAmount
} (if (cmp != 0
؛return cmp
{
؛(return this.salesRetailer.compareTo(o.salesRetailer
{

Override@
} ()public String toString
؛()return salesRetailer.toString() + ", " + salesAmount.toString
{

public static class TokenizerMapper extends Mapper<Object, Text, SalesRetailer,
} <DoubleWritable
؛()private SalesRetailer sales = new SalesRetailer

public void map(Object key, Text value, Context context) throws IOException,
} InterruptedException
؛("،")String[] fields = value.toString().split
```

```
} (if (fields.length == 6

؛()String retailer = fields[0].trim

؛()String city = fields[2].trim

؛((""  ،"$")double pricePerUnit = Double.parseDouble(fields[4].trim().replace

؛((""  ،"،")int unitsSold = Integer.parseInt(fields[5].trim().replace

؛double totalSales = pricePerUnit * unitsSold

؛((sales.setSalesRetailer(new Text(retailer + ", " + city

؛((sales.setSalesAmount(new DoubleWritable(totalSales

؛((context.write(sales, new DoubleWritable(totalSales

{

{

{


public static class SumReducer extends Reducer<SalesRetailer, DoubleWritable, Text,
} <DoubleWritable

؛()private DoubleWritable result = new DoubleWritable


(public void reduce(SalesRetailer key, Iterable<DoubleWritable> values, Context context

} throws IOException, InterruptedException

؛double sum = 0.0

} (for (DoubleWritable val : values

؛()sum += val.get

{

؛(result.set(sum

؛(context.write(new Text(key.getSalesRetailer().toString()), result

{

{
} public static class SalesRetailerComparator extends WritableComparator
} ()protected SalesRetailerComparator
؛(super(SalesRetailer.class, true
{

Override@
} (public int compare(WritableComparable w1, WritableComparable w2
؛SalesRetailer k1 = (SalesRetailer) w1
؛SalesRetailer k2 = (SalesRetailer) w2
؛(return k1.compareTo(k2
{
{
} public static void main(String[] args) throws Exception
؛()Configuration conf = new Configuration
؛("Job job = Job.getInstance(conf, "sales retailer
؛(job.setJarByClass(SalesRetailer.class
؛(job.setMapperClass(TokenizerMapper.class
؛(job.setReducerClass(SumReducer.class
؛(job.setMapOutputKeyClass(SalesRetailer.class
؛(job.setMapOutputValueClass(DoubleWritable.class
؛(job.setOutputKeyClass(Text.class
؛(job.setOutputValueClass(DoubleWritable.class
؛(([FileInputFormat.addInputPath(job, new Path(args[0
؛(([FileOutputFormat.setOutputPath(job, new Path(args[1
؛(job.setSortComparatorClass(SalesRetailerComparator.class
؛(System.exit(job.waitForCompletion(true) ? 0 : 1
{
{
```