Mapp

```
package com.mapreduce.teoria;

import java.io.IOException;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;

public class SalesMapper extends Mapper<Object, Text, Text, DoubleWritable> {
```

```
private Text storeHolidayKey = new Text();
```

```
private DoubleWritable salesValue = new DoubleWritable();

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

```
String[] fields = value.toString().split(",");
if (fields.length == 5) {
    String store = fields[0];
    String isHoliday = fields[4];
    double weeklySales = Double.parseDouble(fields[3]);

    // Crear clave compuesta "store-isHoliday"
    storeHolidayKey.set(store + "-" + isHoliday);
    salesValue.set(weeklySales);
    context.write(storeHolidayKey, salesValue);
}
```

```
}
}
```

Reduce

```
package com.mapreduce.teoria;

import java.io.IOException;

import org.apache.hadoop.io.DoubleWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Reducer;
```

```
public class SalesReducer extends Reducer<Text, DoubleWritable, Text, Text> {

public void reduce(Text key, Iterable values, Context context) throws IOException,
InterruptedException {

double sum = 0.0;

int count = 0;

for (DoubleWritable val : values) {

sum += val.get();

count++;

}

double average = sum / count;

String result = String.format(": %.4f €", average);

context.write(key, new Text(result));

}

}
```

Clase principal

```
package com.mapreduce.teoria;

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.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class SalesAverage {
```

```
public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    Job job = Job.getInstance(conf, "sales average");
    job.setJarByClass(SalesAverage.class);
    job.setMapperClass(SalesMapper.class);
    job.setCombinerClass(SalesReducer.class); // Combiner para optimizar
    job.setReducerClass(SalesReducer.class);
    job.setOutputKeyClass(Text.class);
```

```
      job.setOutputValueClass(DoubleWritable.class);
      FileInputFormat.addInputPath(job, new Path(args[0]));
      FileOutputFormat.setOutputPath(job, new Path(args[1]));
      System.exit(job.waitForCompletion(true) ? 0 : 1);
  }
```

}