```
package csv_all;
import java.io.IOException;
//Hadoop-specific classes
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
//MapReduce classes
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
//Input and Output format classes
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
public class csv_all {
// ================================
//         MAPPER CLASS
// ================================
// Input: (LineNumber, LineContent)
// Output: ("Stats", Marks)
public static class StatsMapper extends Mapper<LongWritable, Text, Text,
IntWritable> {

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

        // Skip header row
        if (key.get() == 0 && value.toString().contains("Name")) {
            return;
        }
        // Split CSV line by comma
        String[] fields = value.toString().split(",");
        try {
            // Extract marks (3rd column, index 2)
            int marks = Integer.parseInt(fields[2]);
            // Emit with same key "Stats"
            context.write(new Text("Stats"), new IntWritable(marks));
        } catch (Exception e) {
            // Skip bad data
        }
    }
}
// ================================
//         REDUCER CLASS
// ================================
// Input: ("Stats", list of marks)
// Output: Min, Max, Sum, Average
```

```java
public static class StatsReducer extends Reducer<Text, IntWritable, Text,
DoubleWritable> {

    public void reduce(Text key, Iterable<IntWritable> values, Context context)
throws IOException, InterruptedException {
        int min = Integer.MAX_VALUE;
        int max = Integer.MIN_VALUE;
        int sum = 0;
        int count = 0;
        // Loop through all marks
        for (IntWritable val : values) {
            int mark = val.get();
            if (mark < min) {
                min = mark;
            }
            if (mark > max) {
                max = mark;
            }
            sum += mark;
            count++;
        }
        double average = (count == 0) ? 0.0 : (double) sum / count;
        // Emit results
        context.write(new Text("Minimum"), new DoubleWritable(min));
        context.write(new Text("Maximum"), new DoubleWritable(max));
        context.write(new Text("Sum"), new DoubleWritable(sum));
        context.write(new Text("Average"), new DoubleWritable(average));
    }
}
// ================================
//        DRIVER / MAIN
// ================================
public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    Job job = Job.getInstance(conf, "Statistics Calculation");
    job.setJarByClass(csv_all.class);
    job.setMapperClass(StatsMapper.class);
    job.setReducerClass(StatsReducer.class);
    // Output types
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    // Input / Output paths
    FileInputFormat.addInputPath(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));
    // Run the job
    System.exit(job.waitForCompletion(true) ? 0 : 1);
}
}
```