**DATASET TO BE CONSIDERED**

Date Stock Open High Low Close Volume

2024-01-01 GOOG 1450.00 1465.00 1445.00 1460.00 1200000

2024-01-01 AAPL 300.00 305.00 295.00 304.00 2100000

2024-01-02 GOOG 1460.00 1480.00 1450.00 1470.00 1100000

**Mapper Class**

The Mapper class will extract the stock symbol and closing price from each line.

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Mapper;

import java.io.IOException;

public class StockMapper extends Mapper<LongWritable, Text, Text, Text> {

private Text stockSymbol = new Text();

private Text closePrice = new Text();

@Override

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

String line = value.toString();

String[] fields = line.split(",");

// Skip the header line

if (fields[0].equals("Date")) {

return;

}

String stock = fields[1];

String close = fields[5];

stockSymbol.set(stock);

closePrice.set(close);

context.write(stockSymbol, closePrice);

}

}

**Reducer Class**

The Reducer class will calculate the average closing price for each stock.

import org.apache.hadoop.io.Text;

import org.apache.hadoop.io.DoubleWritable;

import org.apache.hadoop.mapreduce.Reducer;

import java.io.IOException;

public class StockReducer extends Reducer<Text, Text, Text, DoubleWritable> {

@Override

protected void reduce(Text key, Iterable<Text> values, Context context) throws IOException, InterruptedException {

double sum = 0.0;

int count = 0;

for (Text value : values) {

sum += Double.parseDouble(value.toString());

count++;

}

double average = sum / count;

context.write(key, new DoubleWritable(average));

}

}

**Driver Class**

Finally, the Driver class to configure and run the job.

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 StockAnalysis {

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

if (args.length != 2) {

System.err.println("Usage: StockAnalysis <input path> <output path>");

System.exit(-1);

}

Configuration conf = new Configuration();

Job job = Job.getInstance(conf, "Stock Analysis");

job.setJarByClass(StockAnalysis.class);

job.setMapperClass(StockMapper.class);

job.setReducerClass(StockReducer.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);

}

}

**# hadoop jar StockAnalysis.jar StockAnalysis /input/stocks.csv /output/stock\_analysis**

**# Output**

The output will be a list of stock symbols and their average closing prices:

**AAPL 303.5**

**GOOG 1465.0**