**Experiment -5**

**Analysis of stock market data using MapReduce Program**

**Aim:** Analysis of stock market data using MapReduce Program

**Code:**

**MaxClosePrice.java**

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.FloatWritable;

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.input.TextInputFormat;

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

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

public class MaxClosePrice {

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

if (args.length != 2) {

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

System.*exit*(-1);

}

// Define MapReduce job

Job job = new Job();

job.setJarByClass(MaxClosePrice.class);

job.setJobName("MaxClosePrice");

// Set input and output locations

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

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

// Set Input and Output formats

job.setInputFormatClass(TextInputFormat.class);

job.setOutputFormatClass(TextOutputFormat.class);

// Set Mapper and Reduce classes

job.setMapperClass(MaxClosePriceMapper.class);

job.setReducerClass(MaxClosePriceReducer.class);

// Output types

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(FloatWritable.class);

// Submit job

System.*exit*(job.waitForCompletion(true) ? 0 : 1);

}

}

**MaxClosePriceReducer.java**

import java.io.IOException;

import org.apache.hadoop.io.FloatWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Reducer;

public class MaxClosePriceReducer

extends Reducer<Text, FloatWritable, Text, FloatWritable> {

@Override

public void reduce(Text key, Iterable<FloatWritable> values, Context context)

throws IOException, InterruptedException {

float maxClosePrice = Float.*MIN\_VALUE*;

//Iterate all temperatures for a year and calculate maximum

for (FloatWritable value : values) {

maxClosePrice = Math.*max*(maxClosePrice, value.get());

}

//Write output

context.write(key, new FloatWritable(maxClosePrice));

}

}

**MaxClosePriceMapper.java**

import java.io.IOException;

import org.apache.hadoop.io.FloatWritable;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Mapper;

public class MaxClosePriceMapper extends Mapper<LongWritable, Text, Text, FloatWritable> {

@Override

public void map(LongWritable key, Text value, Context context)

throws IOException, InterruptedException {

String line = value.toString();

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

String stock = items[1];

Float closePrice = Float.*parseFloat*(items[6]);

context.write(new Text(stock), new FloatWritable(closePrice));

}

}

**Output :**









