

<b>Ex.No:3</b>  <b>Date:</b>	<b>IMPLEMENT AN MR PROGRAM THAT PROCESSES A WEATHER DATASET</b>
------------------------------------	---

**AIM:**

Map Reduce program to that processes a weather dataset R.

**PROCEDURE:**

1. Analyze the input file content.
2. Develop the code.
  - a. Writing a map function.
  - b. Writing a reduce function.
  - c. Writing the Driver class.
3. Compiling the source.
4. Building the JAR file.
5. Starting the DFS.
6. Creating Input path in HDFS and moving the data into Input path.
7. Executing the program.

**PROGRAM:**

```
import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.output.MultipleOutputs;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

public class CalculateMaxAndMinTemperatureWithTime {
    public static String calOutputName = "California";
    public static String nyOutputName = "Newyork";
```

```

public static String njOutputName = "Newjersy";
public static String ausOutputName = "Austin";
public static String bosOutputName = "Boston";
public static String balOutputName = "Baltimore";
public static class WhetherForecastMapper extends
    Mapper<Object, Text, Text, Text> {
    public void map(Object keyOffset, Text dayReport, Context con)
        throws IOException, InterruptedException {
        StringTokenizer strTokens = new StringTokenizer(
            dayReport.toString(), "\t");
        int counter = 0;
        Float currnetTemp = null;
        Float minTemp = Float.MAX_VALUE;
        Float maxTemp = Float.MIN_VALUE;
        String date = null;
        String currentTime = null;
        String minTempANDTime = null;
        String maxTempANDTime = null;

        while (strTokens.hasMoreElements()) {
            if (counter == 0) {
                date = strTokens.nextToken();
            } else {
                if (counter % 2 == 1) {
                    currentTime = strTokens.nextToken();
                } else {
                    currnetTemp =
Float.parseFloat(strTokens.nextToken());
                    if (minTemp > currnetTemp) {
                        minTemp = currnetTemp;
                        minTempANDTime = minTemp + "AND" +
currentTime;
                    }
                    if (maxTemp < currnetTemp) {
                        maxTemp = currnetTemp;
                        maxTempANDTime = maxTemp + "AND"
+ currentTime;
                    }
                }
            }
            counter++;
        }
        // Write to context - MinTemp, MaxTemp and corresponding time
        Text temp = new Text();
        temp.set(maxTempANDTime);
        Text dateText = new Text();

```

```

        dateText.set(date);
        try {
            con.write(dateText, temp);
        } catch (Exception e) {
            e.printStackTrace();
        }
        temp.set(minTempANDTime);
        dateText.set(date);
        con.write(dateText, temp);
    }
}

public static class WhetherForecastReducer extends
    Reducer<Text, Text, Text, Text> {
    MultipleOutputs<Text, Text> mos;
    public void setup(Context context) {
        mos = new MultipleOutputs<Text, Text>(context);
    }
    public void reduce(Text key, Iterable<Text> values, Context context)
        throws IOException, InterruptedException {
        int counter = 0;
        String reducerInputStr[] = null;
        String f1Time = "";
        String f2Time = "";
        String f1 = "", f2 = "";
        Text result = new Text();
        for (Text value : values) {
            if (counter == 0) {
                reducerInputStr = value.toString().split("AND");
                f1 = reducerInputStr[0];
                f1Time = reducerInputStr[1];
            }
            else {
                reducerInputStr = value.toString().split("AND");
                f2 = reducerInputStr[0];
                f2Time = reducerInputStr[1];
            }
            counter = counter + 1;
        }
        if (Float.parseFloat(f1) > Float.parseFloat(f2)) {
            result = new Text("Time: " + f2Time + " MinTemp: " + f2 + "\t"
                + "Time: " + f1Time + " MaxTemp: " + f1);
        } else {
            result = new Text("Time: " + f1Time + " MinTemp: " + f1 + "\t"
                + "Time: " + f2Time + " MaxTemp: " + f2);
        }
    }
}

```

```

        String fileName = "";
        if (key.toString().substring(0, 2).equals("CA")) {
            fileName = CalculateMaxAndMinTemperature.calOutputName;
        } else if (key.toString().substring(0, 2).equals("NY")) {
            fileName = CalculateMaxAndMinTemperature.nyOutputName;
        } else if (key.toString().substring(0, 2).equals("NJ")) {
            fileName = CalculateMaxAndMinTemperature.njOutputName;
        } else if (key.toString().substring(0, 3).equals("AUS")) {
            fileName = CalculateMaxAndMinTemperature.ausOutputName;
        } else if (key.toString().substring(0, 3).equals("BOS")) {
            fileName = CalculateMaxAndMinTemperature.bosOutputName;
        } else if (key.toString().substring(0, 3).equals("BAL")) {
            fileName = CalculateMaxAndMinTemperature.balOutputName;
        }
        String strArr[] = key.toString().split("_");
        key.set(strArr[1]);
        mos.write(fileName, key, result);
    }
    @Override
    public void cleanup(Context context) throws IOException,
        InterruptedException {
        mos.close();
    }
}

public static void main(String[] args) throws IOException,
    ClassNotFoundException, InterruptedException {
    Configuration conf = new Configuration();
    Job job = Job.getInstance(conf, "Wheather Statistics of USA");
    job.setJarByClass(CalculateMaxAndMinTemperatureWithTime.class);
    job.setMapperClass(WhetherForecastMapper.class);
    // job.setCombinerClass(IntSumReducer.class);
    job.setReducerClass(WhetherForecastReducer.class);
    job.setMapOutputKeyClass(Text.class);
    job.setMapOutputValueClass(Text.class);
    job.setOutputKeyClass(Text.class); // <hadoop,4>
    job.setOutputValueClass(Text.class);
    MultipleOutputs.addNamedOutput(job, calOutputName,
        TextOutputFormat.class, Text.class, Text.class);
    MultipleOutputs.addNamedOutput(job, nyOutputName,
        TextOutputFormat.class, Text.class, Text.class);
    MultipleOutputs.addNamedOutput(job, njOutputName,
        TextOutputFormat.class, Text.class, Text.class);
    MultipleOutputs.addNamedOutput(job, bosOutputName,
        TextOutputFormat.class, Text.class, Text.class);
    MultipleOutputs.addNamedOutput(job, ausOutputName,
        TextOutputFormat.class, Text.class, Text.class);
    MultipleOutputs.addNamedOutput(job, balOutputName,

```

```

        TextOutputFormat.class, Text.class, Text.class);

    // FileInputFormat.addInputPath(job, new Path(args[0]));
    // FileOutputFormat.setOutputPath(job, new Path(args[1]));
    Path pathInput = new Path(

"hdfs://192.168.213.133:54310/wheatherInputData/input_temp.txt");
    Path pathOutputDir = new Path(
        "hdfs://192.168.213.133:54310/user/hduser1/testfs/output_mapred5");
    FileInputFormat.addInputPath(job, pathInput);
    FileOutputFormat.setOutputPath(job, pathOutputDir);
    try {
        System.exit(job.waitForCompletion(true) ? 0 : 1);
        System.out.println("Job executed successfully!!");
    } catch (Exception e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}
}
}

```

### **Input Dataset:**

CA_25-Jan-2014	00:12:345	15.7	01:19:345	23.1	02:34:542	12.3	
03:12:187	16	04:00:093	-14	05:12:345	35.7	06:19:345	23.1
07:34:542	12.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
11:19:345	23.1	12:34:542	-22.3	13:12:187	16	14:00:093	-7
15:12:345	15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	16
22:00:093	-7						
CA_26-Jan-2014	00:54:245	15.7	01:19:345	23.1	02:34:542	12.3	
03:12:187	16	04:00:093	-14	05:12:345	55.7	06:19:345	23.1
07:34:542	12.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
11:19:345	23.1	12:34:542	12.3	13:12:187	16	14:00:093	-7
15:12:345	15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	16
22:00:093	-7						
CA_27-Jan-2014	00:14:045	35.7	01:19:345	23.1	02:34:542	-22.3	
03:12:187	16	04:00:093	-14	05:12:345	35.7	06:19:345	23.1
07:34:542	12.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
11:19:345	23.1	12:34:542	12.3	13:12:187	16	14:00:093	-7
15:12:345	15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	16
22:00:093	-7						
CA_28-Jan-2014	00:22:315	15.7	01:19:345	23.1	02:34:542	12.3	
03:12:187	16	04:00:093	-14	05:12:345	35.7	06:19:345	23.1
07:34:542	12.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
11:19:345	-23.3	12:34:542	12.3	13:12:187	16	14:00:093	-7
15:12:345	15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	16
22:00:093	-7						

CA_29-Jan-2014	00:15:345	15.7	01:19:345	23.1	02:34:542	52.9	
03:12:187	16	04:00:093	-14	05:12:345	45.0	06:19:345	23.1
07:34:542	-2.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
11:19:345	23.1	12:34:542	12.3	13:12:187	16	14:00:093	-17
15:12:345	15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	16
22:00:093	-7						
NJ_29-Jan-2014	00:15:345	15.7	01:19:345	23.1	02:34:542	52.9	
03:12:187	16	04:00:093	-14	05:12:345	45.0	06:19:345	23.1
07:34:542	-2.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
11:19:345	23.1	12:34:542	12.3	13:12:187	16	14:00:093	-17
15:12:345	15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	16
22:00:093	-7						
CA_30-Jan-2014	00:22:445	15.7	01:19:345	23.1	02:34:542	12.3	
03:12:187	56	04:00:093	-14	05:12:345	35.7	06:19:345	39.6
07:34:542	12.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
11:19:345	23.1	12:34:542	12.3	13:12:187	16	14:00:093	-7
15:12:345	-15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	16
22:00:093	-7						
CA_31-Jan-2014	00:42:245	15.7	01:19:345	23.1	02:34:542	12.3	
03:12:187	16	04:00:093	-14	05:12:345	49.2	06:19:345	23.1
07:34:542	12.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
11:19:345	23.1	12:34:542	12.3	13:12:187	16	14:00:093	-7
15:12:345	15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	16
22:00:093	-27						
NY_29-Jan-2014	00:15:345	15.7	01:19:345	23.1	02:34:542	52.9	
03:12:187	16	04:00:093	-14	05:12:345	45.0	06:19:345	23.1
07:34:542	-2.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
11:19:345	23.1	12:34:542	12.3	13:12:187	16	14:00:093	-17
15:12:345	15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	16
22:00:093	-7						
NY_30-Jan-2014	00:22:445	15.7	01:19:345	23.1	02:34:542	12.3	
03:12:187	56	04:00:093	-14	05:12:345	35.7	06:19:345	39.6
07:34:542	12.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
11:19:345	23.1	12:34:542	12.3	13:12:187	16	14:00:093	-7
15:12:345	-15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	16
22:00:093	-7						
NY_31-Jan-2014	00:42:245	15.7	01:19:345	23.1	02:34:542	12.3	
03:12:187	16	04:00:093	-14	05:12:345	49.2	06:19:345	23.1
07:34:542	12.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
11:19:345	23.1	12:34:542	12.3	13:12:187	16	14:00:093	-7
15:12:345	15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	16
22:00:093	-27						
NJ_30-Jan-2014	00:22:445	15.7	01:19:345	23.1	02:34:542	12.3	
03:12:187	56	04:00:093	-14	05:12:345	35.7	06:19:345	39.6
07:34:542	12.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
11:19:345	23.1	12:34:542	12.3	13:12:187	16	14:00:093	-7

	15:12:345	-15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	16
	22:00:093	-7						
AUS_25-Jan-2014	00:12:345	15.7	01:19:345	23.1	02:34:542	12.3		
	03:12:187	16	04:00:093	-14	05:12:345	35.7	06:19:345	23.1
	07:34:542	12.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
	11:19:345	23.1	12:34:542	-22.3	13:12:187	16	14:00:093	-7
	15:12:345	15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	16
	22:00:093	-7						
AUS_26-Jan-2014	00:54:245	15.7	01:19:345	53.1	02:34:542	12.3		
	03:12:187	16	04:00:093	-14	05:12:345	55.7	06:19:345	23.1
	07:34:542	12.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
	11:19:345	23.1	12:34:542	12.3	13:12:187	16	14:00:093	-7
	15:12:345	15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	16
	22:00:093	-27						
AUS_27-Jan-2014	00:14:045	35.7	01:19:345	23.1	02:34:542	-22.3		
	03:12:187	16	04:00:093	-14	05:12:345	55.7	06:19:345	23.1
	07:34:542	12.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
	11:19:345	23.1	12:34:542	12.3	13:12:187	16	14:00:093	-7
	15:12:345	15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	16
	22:00:093	-7						
BOS_28-Jan-2014	00:22:315	15.7	01:19:345	33.1	02:34:542	12.3		
	03:12:187	16	04:00:093	-14	05:12:345	35.7	06:19:345	23.1
	07:34:542	12.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
	11:19:345	-23.3	12:34:542	12.3	13:12:187	16	14:00:093	-7
	15:12:345	15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	16
	22:00:093	-7						
AUS_29-Jan-2014	00:15:345	15.7	01:19:345	23.1	02:34:542	62.9		
	03:12:187	16	04:00:093	-14	05:12:345	45.0	06:19:345	23.1
	07:34:542	-2.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
	11:19:345	23.1	12:34:542	12.3	13:12:187	16	14:00:093	-17
	15:12:345	15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	36
	22:00:093	-7						
BOS_29-Jan-2014	00:15:345	15.7	01:19:345	23.1	02:34:542	52.9		
	03:12:187	16	04:00:093	-14	05:12:345	45.0	06:19:345	23.1
	07:34:542	-2.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
	11:19:345	23.1	12:34:542	12.3	13:12:187	16	14:00:093	-17
	15:12:345	15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	16
	22:00:093	-7						
BOS_30-Jan-2014	00:22:445	15.7	01:19:345	13.1	02:34:542	12.3		
	03:12:187	56	04:00:093	-14	05:12:345	35.7	06:19:345	39.6
	07:34:542	12.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
	11:19:345	23.1	12:34:542	12.3	13:12:187	16	14:00:093	-7
	15:12:345	-15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	16
	22:00:093	-7						
BAL_29-Jan-2014	00:42:245	15.7	01:19:345	23.1	02:34:542	12.3		
	03:12:187	16	04:00:093	-14	05:12:345	49.2	06:19:345	23.1

	07:34:542	12.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
	11:19:345	23.1	12:34:542	12.3	13:12:187	16	14:00:093	-27
	15:12:345	15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	16
	22:00:093	-27						
BAL_30-Jan-2014	00:15:345	15.7	01:19:345	23.1	02:34:542	52.9		
	03:12:187	16	04:00:093	-14	05:12:345	45.0	06:19:345	23.1
	07:34:542	-2.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
	11:19:345	23.1	12:34:542	12.3	13:12:187	16	14:00:093	-17
	15:12:345	15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	16
	22:00:093	-7						
BAL_31-Jan-2014	00:22:445	15.7	01:19:345	13.1	02:34:542	12.3		
	03:12:187	56	04:00:093	-14	05:12:345	35.7	06:19:345	39.6
	07:34:542	12.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
	11:19:345	23.1	12:34:542	12.3	13:12:187	16	14:00:093	-7
	15:12:345	-15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	16
	22:00:093	-7						
AUS_30-Jan-2014	00:42:245	15.7	01:19:345	23.1	02:34:542	12.3		
	03:12:187	16	04:00:093	-14	05:12:345	49.2	06:19:345	23.1
	07:34:542	12.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
	11:19:345	23.1	12:34:542	12.3	13:12:187	16	14:00:093	-7
	15:12:345	15.7	16:19:345	23.1	19:34:542	12.3	20:12:187	16
	22:00:093	-27						
AUS_31-Jan-2014	00:22:445	15.7	01:19:345	43.1	02:34:542	12.3		
	03:12:187	56	04:00:093	-14	05:12:345	35.7	06:19:345	39.6
	07:34:542	12.3	08:12:187	16	09:00:093	-7	10:12:345	15.7
	11:19:345	23.1	12:34:542	12.3	13:12:187	16	14:00:093	-17
	15:12:345	-15.7	16:19:345	23.1	19:34:542	32.3	20:12:187	16
	22:00:093	-17						

## **RESULT:**

Thus the Map Reduce program that processes a weather dataset R is executed successfully.



## **OUTPUT:**

```
hduser1@ubuntu:/usr/local/hadoop2.6.1/bin$ ./hadoop fs -cat
/user/hduser1/testfs/output_mapred3/Austin-r-00000
25-Jan-2018 Time: 12:34:542 MinTemp: -22.3 Time: 05:12:345 MaxTemp: 35.7
26-Jan-2018 Time: 22:00:093 MinTemp: -27.0 Time: 05:12:345 MaxTemp: 55.7
27-Jan-2018 Time: 02:34:542 MinTemp: -22.3 Time: 05:12:345 MaxTemp: 55.7
29-Jan-2018 Time: 14:00:093 MinTemp: -17.0 Time: 02:34:542 MaxTemp: 62.9
30-Jan-2018 Time: 22:00:093 MinTemp: -27.0 Time: 05:12:345 MaxTemp: 49.2
31-Jan-2018 Time: 14:00:093 MinTemp: -17.0 Time: 03:12:187 MaxTemp: 56.0
```