Exp No: 11 **Date:**

HADOOP
IMPLEMENT THE MAX TEMPERATURE MAPREDUCE PROGRAM TO IDENTIFY THE YEAR WISE MAXIMUM TEMPERATURE FROM SENSOR DATA

AIM

To implement the Max temperature MapReduce program to identify the year-wise maximum temperature from the sensor data.

Description

Sensors senses weather data in big text format containing station ID, year, date, time, temperature, quality etc. from each sensor and store it in a single line. Suppose thousands of data sensors are there, then we have thousands of records with no particular order. We require only a year and maximum temperature of particular quality in that year.

For example:

Input string from sensor:
002902907099999*1902*010720004+64333+023450
FM-12+
000599999V0202501N027819999999N0000001N9-*00331*+
99999098351ADDGF102991999999999999999999

Here: 1902 is year

0033 is temperature

1 is measurement quality (Range between 0 or 1 or 4 or 5 or 9)

Here each mapper takes the input key as "byte offset of line" and value as "one weather sensor read i.e one line". and parse each line and produce an intermediate key "year" and intermediate value as "temperature of certain measurement qualities" for that year.

The combiner will form set values of temperature. Year and set of values of temperatures is given as input <key, value> to reducer and Reducer will produce year and maximum temperature for that year from the set of temperature values.

PROGRAM

*/

```
import  org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
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;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
importorg.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
importorg.apache.hadoop.mapreduce.Reducer;

//Mapper class


class MaxTemperatureMapper
extends Mapper<LongWritable, Text, Text, IntWritable> { private static final int MISSING

= 9999;

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

String line = value.toString(); String year = line.substring(15, 19); int airTemperature;
if (line.charAt(87) == '+') { // parseInt doesn't like leading plus signs airTemperature =
Integer.parseInt(line.substring(88, 92));
} else {
airTemperature = Integer.parseInt(line.substring(87, 92));
}
String quality = line.substring(92, 93);
if (airTemperature != MISSING && quality.matches("[01459]")) { context.write(new
Text(year), new IntWritable(airTemperature));
}
}
}

//Reducer class
class MaxTemperatureReducer
extends Reducer<Text, IntWritable, Text, IntWritable> {

@Override
public void reduce(Text key, Iterable<IntWritable> values, Context context)
throws IOException, InterruptedException {
```

```
int maxValue = Integer.MIN_VALUE; for (IntWritable value : values) {
maxValue = Math.max(maxValue, value.get());
}
context.write(key, new IntWritable(maxValue));
}
}
//Driver Class

public class MaxTemperature {

public static void main(String[] args) throws Exception { if (args.length != 2) {
System.err.println("Usage: MaxTemperature <input path=""> <output path>"); System.exit(-
1);
}

Job job = Job.getInstance(new Configuration()); job.setJarByClass(MaxTemperature.class);
job.setJobName("Max temperature");

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


job.setMapperClass(MaxTemperatureMapper.class);
job.setReducerClass(MaxTemperatureReducer.class);

job.setOutputKeyClass(Text.class); job.setOutputValueClass(IntWritable.class);

job.submit();
}
}
```

OUTPUT:

Input for String :

0029029070999991902010720004+64333+023450FM-12+
000599999V0202501N027819999999N0000001N9-00331+
99999098351ADDGF10299199999999999999’

Open ▾ dataset.txt ~/exp3

```
23907 20150101  2.423  -98.08   30.62     2.2    -0.6     0.8     0.9     7.0     1.47 C     3.7     1.1     2.5
99.9    85.4    97.2   0.369   0.308 -99.000 -99.000 -99.000     7.0     8.1 -9999.0 -9999.0 -9999.0
23907 20150102  2.423  -98.08   30.62     3.5     1.3     2.4     2.2    10.2     1.43 C     4.9     2.3     3.1
100.0    98.8    99.8   0.391   0.327 -99.000 -99.000 -99.000     7.1     7.9 -9999.0 -9999.0 -9999.0
23907 20150103  2.423  -98.08   30.62    15.9     2.3     9.1     7.5     3.1    11.00 C    16.4     2.9     7.3
100.0    34.8    73.7   0.450   0.397 -99.000 -99.000 -99.000     7.6     7.9 -9999.0 -9999.0 -9999.0
23907 20150104  2.423  -98.08   30.62     9.2    -1.3     3.9     4.2     0.0    13.24 C    12.4    -0.5     4.9
82.0    40.6    61.7   0.413   0.352 -99.000 -99.000 -99.000     7.3     7.9 -9999.0 -9999.0 -9999.0
23907 20150105  2.423  -98.08   30.62    10.9    -3.7     3.6     2.6     0.0    13.37 C    14.7    -3.0     3.8
77.9    33.3    57.4   0.399   0.340 -99.000 -99.000 -99.000     6.3     7.0 -9999.0 -9999.0 -9999.0
23907 20150106  2.423  -98.08   30.62    20.2     2.9    11.6    10.9     0.0    12.90 C    22.0     1.6     9.9
67.7    30.2    49.3   0.395   0.335 -99.000 -99.000 -99.000     8.0     8.0 -9999.0 -9999.0 -9999.0
23907 20150107  2.423  -98.08   30.62    10.9    -3.4     3.8     4.5     0.0    12.68 C    12.4    -2.1     5.5
82.7    36.5    55.7   0.387   0.328 -99.000 -99.000 -99.000     7.6     8.3 -9999.0 -9999.0 -9999.0
23907 20150108  2.423  -98.08   30.62     0.6    -7.9    -3.6    -3.3     0.0     4.98 C     3.9    -4.8    -0.5
57.7    37.6    48.1   0.372   0.316 -99.000 -99.000 -99.000     4.7     6.1 -9999.0 -9999.0 -9999.0
23907 20150109  2.423  -98.08   30.62     2.0     0.1     1.0     0.8     0.0     2.52 C     4.1     1.2     2.5
87.8    48.9    64.4   0.368   0.312 -99.000 -99.000 -99.000     5.4     6.2 -9999.0 -9999.0 -9999.0
23907 20150110  2.423  -98.08   30.62     0.5    -2.0    -0.8    -0.6     3.9     2.11 C     2.5    -0.1     1.4
99.9    47.7    85.8   0.373   0.314 -99.000 -99.000 -99.000     5.1     6.0 -9999.0 -9999.0 -9999.0
23907 20150111  2.423  -98.08   30.62    10.9     0.0     5.4     4.4     2.6     6.38 C    12.7     1.3     5.8
```

GNU nano 7.2 mapper.py

```
#!/usr/bin/env python
import sys
# input comes from STDIN (standard input)
# the mapper will get daily max temperature and group it by month. so output w>
#(month,dailymax_temperature)

for line in sys.stdin:
        # remove leading and trailing whitespace
        line = line.strip()
        # split the line into words
        words = line.split()
        #See the  README hosted on the weather website which help us understan>
        month = line[10:12]
        daily_max = line[38:45]
        daily_max = daily_max.strip()
        # increase counters
        for word in words:
                # write the results to STDOUT (standard output);
                # what we output here will be go through the shuffle proess an>
                # be the input for the Reduce step, i.e. the input for reducer>
                #
                # tab-delimited; month and daily max temperature as output
                print ('%s\t%s' % (month ,daily_max))
```

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify

```
#!/usr/bin/env python
from operator import itemgetter
import sys
current_month = None
current_max = 0
month = None
for line in sys.stdin:
        line = line.strip()
        month, daily_max = line.split('\t', 1)
        try:
                daily_max = float(daily_max)
        except ValueError:
                continue
        if current_month == month:
                if daily_max > current_max:
                        current_max = daily_max
        else:
                if current_month:
                        print ('%s\t%s' % (current_month, current_max))
                current_max = daily_max
                current_month = month
if current_month == month:
        print ('%s\t%s' % (current_month, current_max))

^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify
```

```
kaarokki@fedora:~$ hadoop jar $HADOOP_STREAMING -input /exp2/dataset.txt -output /exp2/output2 -mapper ~/exp3/mapper.py -reduce
r ~/exp3/reducer.py
packageJarJar: [/tmp/hadoop-unjar3250782673972055567/] [] /tmp/streamjob2672168064368820464.jar tmpDir=null
2024-10-20 11:35:48,890 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2024-10-20 11:35:49,151 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2024-10-20 11:35:49,763 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/kaarokk
i/.staging/job_1729438171769_0003
2024-10-20 11:35:50,157 INFO mapred.FileInputFormat: Total input files to process : 1
2024-10-20 11:35:50,276 INFO mapreduce.JobSubmitter: number of splits:2
2024-10-20 11:35:50,551 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1729438171769_0003
2024-10-20 11:35:50,552 INFO mapreduce.JobSubmitter: Executing with tokens: []
2024-10-20 11:35:50,787 INFO conf.Configuration: resource-types.xml not found
2024-10-20 11:35:50,787 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2024-10-20 11:35:51,415 INFO impl.YarnClientImpl: Submitted application application_1729438171769_0003
2024-10-20 11:35:51,496 INFO mapreduce.Job: The url to track the job: http://fedora:8088/proxy/application_1729438171769_0003/
2024-10-20 11:35:51,502 INFO mapreduce.Job: Running job: job_1729438171769_0003
2024-10-20 11:36:34,950 INFO mapreduce.Job: Job job_1729438171769_0003 running in uber mode : false
2024-10-20 11:36:34,957 INFO mapreduce.Job:  map 0% reduce 0%
2024-10-20 11:37:32,056 INFO mapreduce.Job:  map 50% reduce 0%
2024-10-20 11:37:34,088 INFO mapreduce.Job:  map 100% reduce 0%
2024-10-20 11:37:39,203 INFO mapreduce.Job:  map 100% reduce 100%
2024-10-20 11:37:40,235 INFO mapreduce.Job: Job job_1729438171769_0003 completed successfully
2024-10-20 11:37:40,380 INFO mapreduce.Job: Counters: 54
```

```
        File System Counters
                FILE: Number of bytes read=102094
                FILE: Number of bytes written=1039204
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=83480
                HDFS: Number of bytes written=96
                HDFS: Number of read operations=11
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
                HDFS: Number of bytes read erasure-coded=0
        Job Counters
                Launched map tasks=2
                Launched reduce tasks=1
                Data-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=62981
                Total time spent by all reduces in occupied slots (ms)=5165
                Total time spent by all map tasks (ms)=62981
                Total time spent by all reduce tasks (ms)=5165
                Total vcore-milliseconds taken by all map tasks=62981
                Total vcore-milliseconds taken by all reduce tasks=5165
                Total megabyte-milliseconds taken by all map tasks=64492544
                Total megabyte-milliseconds taken by all reduce tasks=5288960
        Map-Reduce Framework
```

```
kaarokki@fedora:~$ hdfs dfs -cat /exp2/output/part-00000
01      26.5
02      26.6
03      29.1
04      30.8
05      31.1
06      33.6
07      38.5
08      40.2
09      36.5
10      36.9
11      27.6
12      25.9
```

RESULT

Thus a java program has been implemented to identify the year-wise maximum temperature from the sensor data.

Sample Questions

BASIC UNDERSTANDING: Exp 1

1. What is virtualization?

Ans. Virtualization is an abstraction layer that decouples physical hardware from operating system to deliver greater IT resource utilization and flexibility.

2. What is the Difference between Full Virtualization and Para Virtualization?

Ans. Full virtualization & Para virtualization both comes under the Hardware virtualization. Some of the differences between them are listed below:

Full Virtualization: In full virtualization guest VMs (Virtual Machines) are not aware that they are in virtualized environment there-fore the guest os issues command to what it thinks as actual hardware but actually are just simulated devices created by the hosts.

Para Virtualization : In para virtualization the guest vm is aware that it is in a virtualized environment . If guest vm requires resources , it issues commands to host operating system instead of directly communicating with simulated hardware.

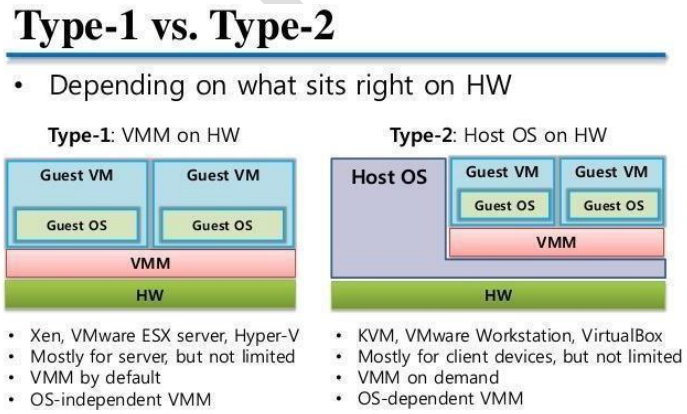
3. What is Hyper-visor ?

A **hypervisor** or virtual machine monitor (VMM) is computer software, firmware or hardware that creates and runs virtual machines. A computer on which a **hypervisor** runs one or more virtual machines is called a host machine, and each virtual machine is called a guest machine.

4. Whatare the difference between Type 1 and Type 2 Hypervisor ?

Ans. Type 1: When the Hypervisor is installed on bare metal / Physical hardware it is known as Type 1 Hypervisor . Examples are VM ware ESXi, Oracle VM, Microsoft Hyper V.

Type 2: When the Hypervisor is installed on top of an operating system it is known as Type 2 Hypervisor . Examples are Microsoft Virtual Server, VM Ware Server and workstation.



BASIC UNDERSTANDING: Exp 2

1. **What is a virtual block?**

   A virtual block device is an interface with applications that appears to the applications as a memory device, such as a standard block device.

2. **What is a virtual disk?**

   Virtual disks are stored as files on the host computer or on a network file server. It does not matter whether the physical disk that holds the files is IDE or SCSI.

   IDE (Integrated Drive Electronics) SCSI(Small Computer System Interface) SATA(Serial Advanced Technology Attachment)

3. **What is a VM clone?**

   A clone is a copy of an existing virtual machine.

4. **What is a Snapshot and a Template?**

   A snapshot is a copy of the virtual machine's disk file at a given point in time. **Snapshots** provide a change log for the virtual disk and are used to restore a VM to a particular point in time when a failure or system error occurs.

   A **template** is a master copy of a virtual machine that can be used to create many clones.