18MCA52- Big Data Analytics

Arpita Manoj Chikkodi

1RV18MCA07

Lab Program - 2

Map Reduce Program using Temperature Dataset

a) Write a Java program for finding Maximum recorded temperature by the year from Weather Dataset

b) Submit the job to cluster

c) Find the status of the Job and terminate it

=>

Dataset Used

NCDC(National Climatic Data Center) Dataset. The 10 files of the year 1990 are merged to get the sample dataset and is named as Sample.txt

Java MapReduce Program to find Maximum Recorded Temperature

1.MaxTemperatureMapper.java

```
package arpita.wd;

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Mapper;

public 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) == '+') {

                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));}

} }
```

2.MaxTemperatureReducer.java

```
package arpita.wd;

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Reducer;

public 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));
} }
```

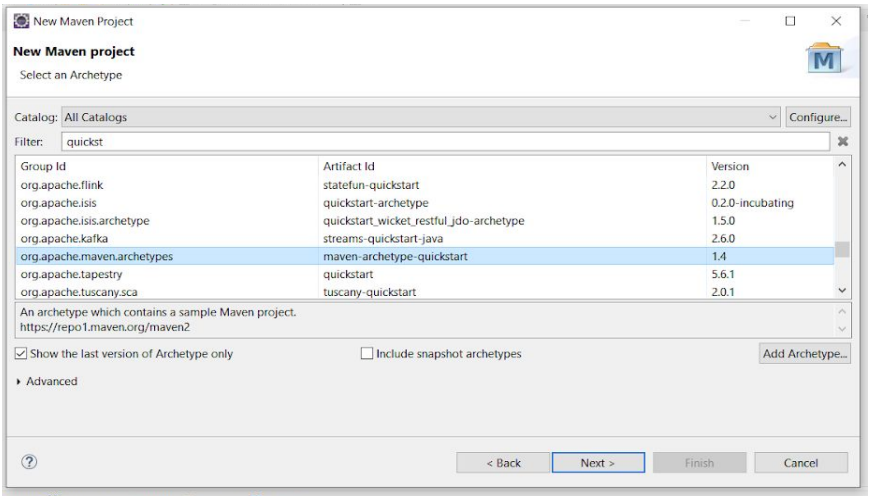
3. MaxTemperature.java

```
package arpita.wd;
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;
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); }
		try(@SuppressWarnings("deprecation") Job job = new Job()) {
			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);

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

        finally {}} }
```

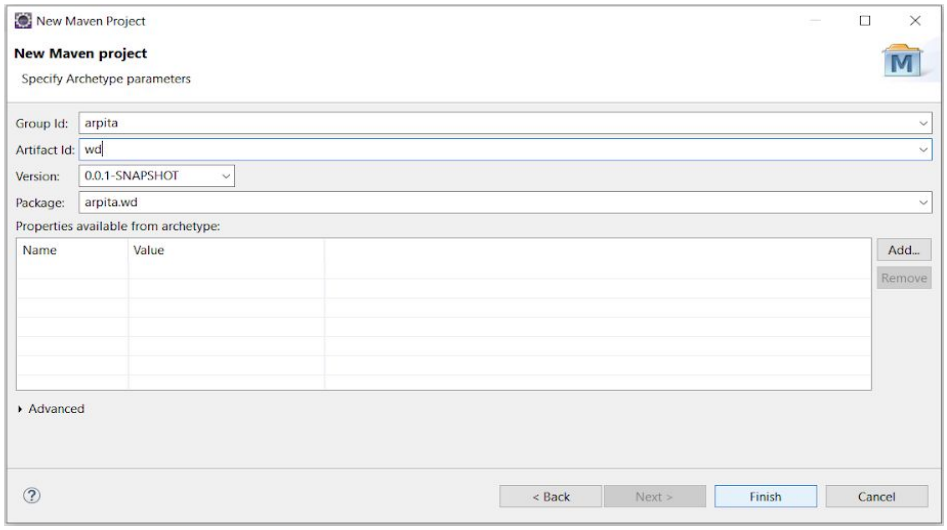
Detailed Steps

Step 1: Create new Maven Project in Eclipse



Step 2: Specify the archetype parameters Group Id and Artifact Id

Group Id - arpita, Artifact Id - wd, package - arpita.wd



Step 3: Create 3 Java classes inside src/main/java namely MaxTemperatureMapper.java, MaxTemperatureReducer.java, MaxTemperature.java

Step 4: Add hadoop-client and hadoop-common dependencies in wd/pom.xml inside <dependencies></dependencies> section

Dependencies

<!-- https://mvnrepository.com/artifact/org.apache.hadoop/hadoop-client -->

<dependency>

<groupId>org.apache.hadoop</groupId>

<artifactId>hadoop-client</artifactId>

<version>3.2.1</version>

</dependency>

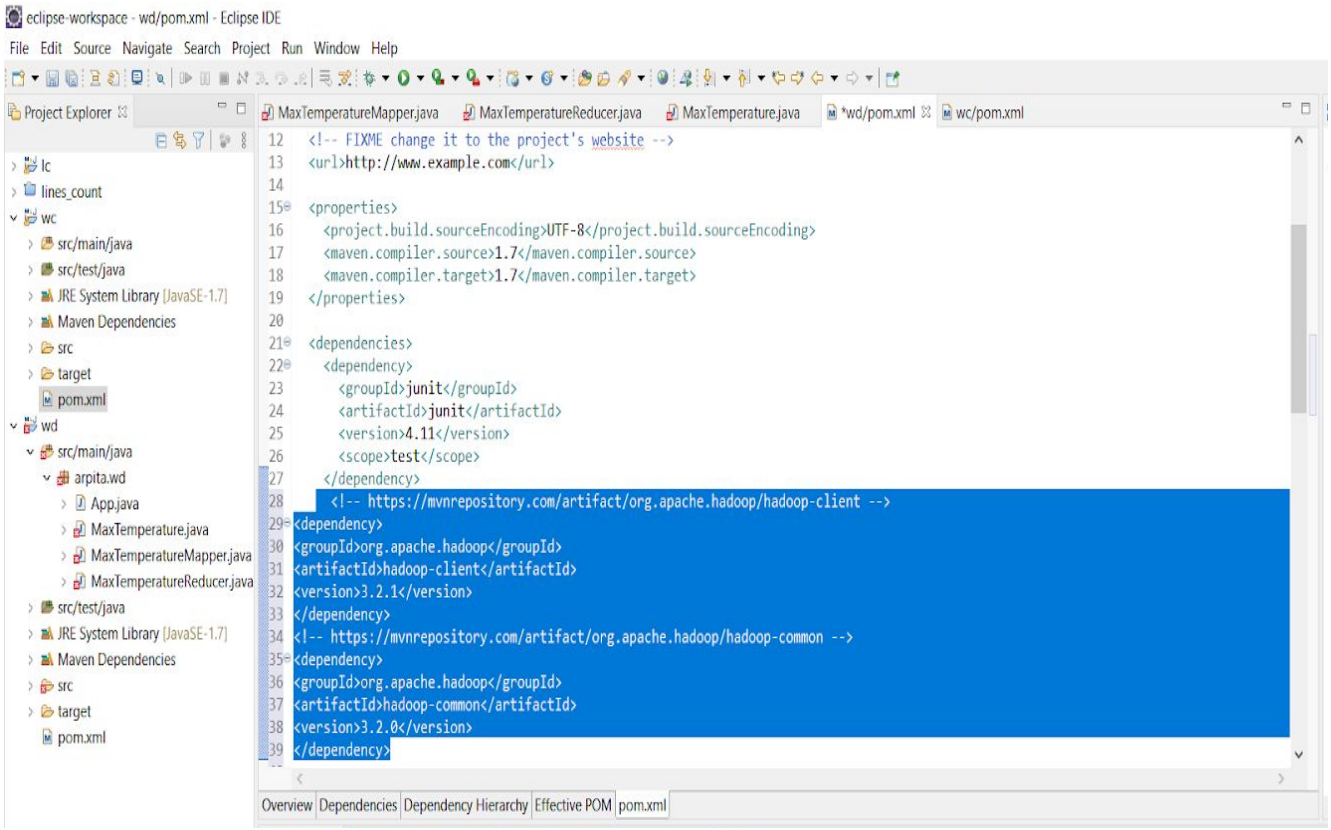
<!-- https://mvnrepository.com/artifact/org.apache.hadoop/hadoop-common -->

<dependency>

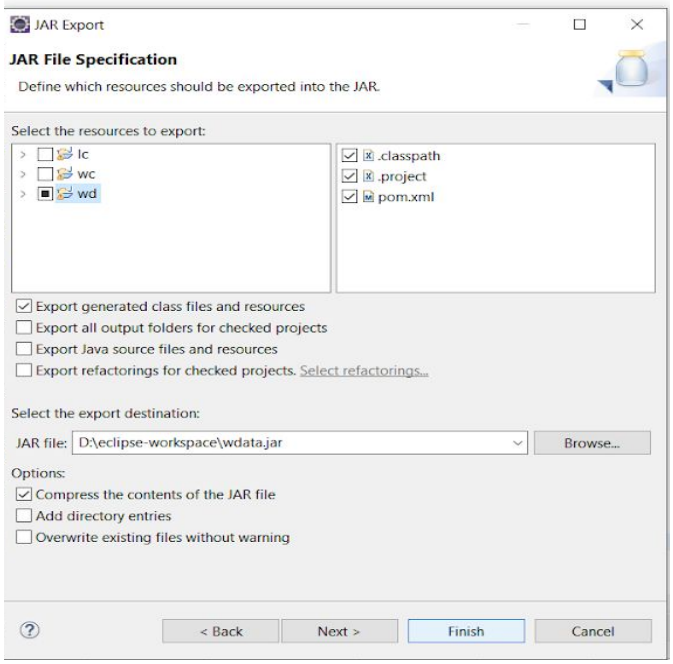
<groupId>org.apache.hadoop</groupId>

<artifactId>hadoop-common</artifactId>

<version>3.2.0</version> </dependency>



Step 5: Save the files and export the package wd as jar with following jar specifications, jar is named as wdata.jar



Step 6: Copy wdata.jar and Sample.txt to /MCA_LAB/lab2/ folder to HDFS through Ambari

Step 7: Copy wdata.jar and Sample.txt from /MCA_LAB/lab2/ to hadoop root/lab2 folder using copyToLocal command by creating a new directory lab2

> mkdir lab2

>hdfs dfs -copyToLocal /MCA_LAB/lab2/Sample.txt lab2/Sample.txt

>hdfs dfs -copyToLocal /MCA_LAB/lab2/wdata.jar lab2/wdata.jar

```
[root@sandbox ~]# mkdir lab2
[root@sandbox ~]# ls
anaconda-ks.cfg  build.out  install.log         lab2        lc_input.txt  sandbox.info     start_hbase.sh
blueprint.json   hdp        install.log.syslog  lcfinal.jar  lcl.jar       start_ambari.sh
[root@sandbox ~]# hdfs dfs -copyToLocal /MCA_LAB/lab2/Sample.txt lab2/Sample.txt
[root@sandbox ~]# ls lab2/
Sample.txt
[root@sandbox ~]# hdfs dfs -copyToLocal /MCA_LAB/lab2/wdata.jar lab2/wdata.jar
[root@sandbox ~]# hdfs dfs -copyToLocal /MCA_LAB/lab2/wdata2.jar lab2/wdata2.jar
[root@sandbox ~]# ls lab2/
Sample.txt  wdata2.jar  wdata.jar
```

Step 8: Run the wdata.jar file using the following command
> hadoop jar <jar filename> <classname> <input filename with path> <output filename>

=> hadoop jar lab2/wdata.jar arpita.wd.MaxTemperature /MCA_LAB/lab2/Sample.txt /lab2/wdata_output

```
[root@sandbox ~]# hadoop jar lab2/wdata.jar arpita.wd.MaxTemperature /MCA_LAB/lab2/Sample.txt /lab2/wdata_output1
20/09/25 21:47:25 INFO impl.TimelineClientImpl: Timeline service address: http://sandbox.hortonworks.com:8188/ws/v1/timeline/
20/09/25 21:47:25 INFO client.RMProxy: Connecting to ResourceManager at sandbox.hortonworks.com/172.17.0.2:8050
20/09/25 21:47:25 INFO client.AHSProxy: Connecting to Application History server at sandbox.hortonworks.com/172.17.0.2:10200
20/09/25 21:47:26 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with T
oolRunner to remedy this.
20/09/25 21:47:29 INFO input.FileInputFormat: Total input paths to process : 1
20/09/25 21:47:29 INFO lzo.GPLNativeCodeLoader: Loaded native gpl library
20/09/25 21:47:29 INFO lzo.LzoCodec: Successfully loaded & initialized native-lzo library [hadoop-lzo rev 7a4b57bedce694048432dd5bf5b90a6c8ccdba80]
20/09/25 21:47:30 INFO mapreduce.JobSubmitter: number of splits:1
20/09/25 21:47:36 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1601046793397_0011
20/09/25 21:47:40 INFO impl.YarnClientImpl: Submitted application application_1601046793397_0011
20/09/25 21:47:41 INFO mapreduce.Job: The url to track the job: http://sandbox.hortonworks.com:8088/proxy/application_1601046793397_0011/
20/09/25 21:47:41 INFO mapreduce.Job: Running job: job_1601046793397_0011
20/09/25 21:50:03 INFO mapreduce.Job: Job job_1601046793397_0011 running in uber mode : false
20/09/25 21:50:03 INFO mapreduce.Job:  map 0% reduce 0%
20/09/25 21:51:07 INFO mapreduce.Job:  map 100% reduce 0%
20/09/25 21:51:37 INFO mapreduce.Job:  map 100% reduce 100%
20/09/25 21:51:44 INFO mapreduce.Job: Job job_1601046793397_0011 completed successfully
20/09/25 21:51:45 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=529755
                FILE: Number of bytes written=1347929
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=9168254
                HDFS: Number of bytes written=9
                HDFS: Number of read operations=6
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=57256
                Total time spent by all reduces in occupied slots (ms)=25445
                Total time spent by all map tasks (ms)=57256
                Total time spent by all reduce tasks (ms)=25445
                Total vcore-milliseconds taken by all map tasks=57256
```

Step 9: Output can be displayed with following command
>hdfs dfs -ls /lab2/wdata_output1
>hdfs dfs -cat /lab2/wdata_output1/*

```
[root@sandbox ~]# hdfs dfs -ls /lab2/wdata_output1
Found 2 items
-rw-r--r--   1 root hdfs          0 2020-09-25 21:51 /lab2/wdata_output1/_SUCCESS
-rw-r--r--   1 root hdfs          9 2020-09-25 21:51 /lab2/wdata_output1/part-r-00000
[root@sandbox ~]# hdfs dfs -cat /lab2/wdata_output1
cat: `/lab2/wdata_output1': Is a directory
[root@sandbox ~]# hdfs dfs -cat /lab2/wdata_output1/*
1990    240
[root@sandbox ~]#
```

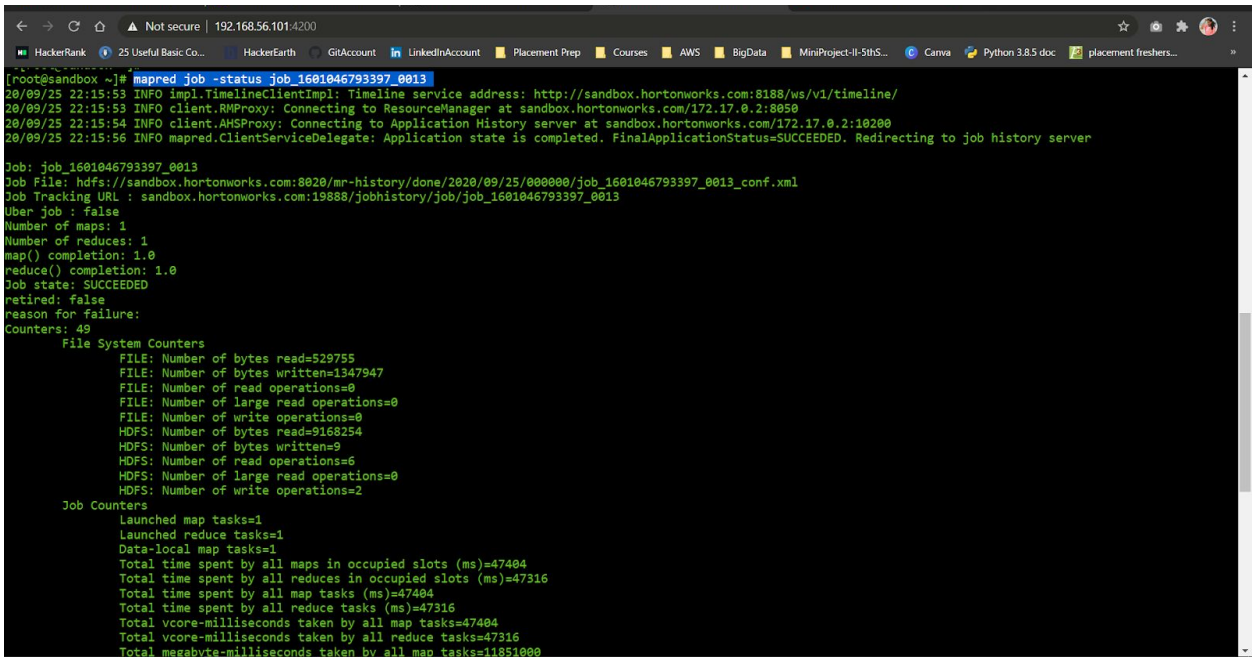
Output Obtained

=>1990 240

So the maximum temperature recorded is 24.0º C for the year 1990

Tracking the status of the Job

> mapred job -status job_id



Killing/Terminating the Job

>mapred job -kill job_id