Practical grading using Rubrics

|  |  |  |  |
| --- | --- | --- | --- |
| **Name of the student:** | Yash Sankpal | **Roll No.** | 8895 |
| **Practical Number:** | 3 | **Date of Practical:** |  |
| **Relevant CO’s** | **At the end of the course students will be able to use tools like hadoop and NoSQL to solve big data related problems.** | | |
| **Sign here to indicate that you have read all the relevant material provided**  **before attempting this practical** | | | **Sign:** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Indicator** | **Very Poor** | **Poor** | **Average** | **Good** | **Excellent** |
| **Timeline**  (2) | More than a session late  (0) | NA | NA | NA | Early or on time (2) |
| **Code de- sign** (2) | N/A | Very poor code design with no comments and indenta-  tion(0.5) | Poor code design with very com-  ments and  indentation (1) | Design with good coding standards (1.5) | Accurate design  with bet-  ter coding satndards (2) |
| **Performance**  (4) | Unable to perform the experiment (0) | Able to partially perform the experiment (1) | Able to perform the experiment for certain use cases (2) | Able to perform the experiment considering most of the  use cases (3) | Able to perform the experiment considering all use cases  (4) |
| **Postlab** (2) | No Execu- tion(0) | N/A | Partially Exe- cuted (1) | N/A | Fully Ex- ecuted  (2) |

|  |  |
| --- | --- |
| **Total Marks (10)** | **Sign of instructor** |
|  |  |

**Practical**

Course title: Big Data Analytics Course term: 2021-2022

Instructor name: Saurabh Kulnarni

Problem Statement: Find year-wise maximum temperature from the given dataset using

map-reduce.

Theory:Explain the working of finding maximum temperature using map reduce with small

example and diagrams

The input is a series of well formatted lines containing temperature and quality data at specific character indices delimited by newlines with data for each year separated by files.

We use the lines to get year, temperature (positive or negative) and the quality code using built-in string functions and validations as per given conditions.

The mapper function returns key as the year of record and value as temperatures recorded.

The reducer function iterates through temperatures recorded per year and returning the max

present value.

Eg:

Mapper Input:

File1:

0029029070999991901010106004+64333+023450FM-12+000599999V0202701N015919999999N0000001N9-00781+99999102001ADDGF108991999999999999999999

0029029070999991901010113004+64333+023450FM-12+000599999V0202901N008219999999N0000001N9-00721+99999102001ADDGF104991999999999999999999

0029029070999991901010120004+64333+023450FM-12+000799999V0209991C000019999999N0000001N9-00941+99999102001ADDGF108991999999999999999999

File2:

0035029070999991902010106004+64333+023450FM-12+000599999V0201401N011819999999N0000001N9-00941+00099100551ADDGF104991999999999999999999MW1381

0035029070999991902010113004+64333+023450FM-12+000599999V0201401N011819999999N0000001N9-01001+00149100311ADDGF104991999999999999999999MW1381

0035029070999991902010120004+64333+023450FM-12+000599999V0201401N013919999999N0000001N9-01171+99999100121ADDGF108991999999999999999999MW1381

Mapper Working:

[1901, 5], [1901, 7], [1902, 9], [1902, 14]

Mapper Output:

[1901, [5, 7]], [1902, [9, 14]]

Reducer Output:

[1901, 7], [1902, 14]

|  |
| --- |
| **Code:**  **code for mapper:**  **public** **class** MaxTemperatureMapper **extends**  Mapper<LongWritable, Text, Text, IntWritable> {  //declare a public static final integer variable for assigning missing value and assign value 9999 as a missing value  **public** **static** **final** **int** ***MISSING*** = 9999;  **public** **void** map(LongWritable ikey, Text ivalue, Context context)  **throws** IOException, InterruptedException {  //Convert value given to map function to string  String val = ivalue.toString();    //extract year(4 digits)which is at 16th position of each record in given data set  String year = val.substring(15, 19);    //define integer variable for measuring air temperature  **int** temp = ***MISSING***;    //check the sign of temperature at 88th position of record. If it is +, then extract only integer value of temperature(4 digits).  //If sign is -, then extract integer value of temperature(4 digit) along with - sign.  **if**(val.charAt(87) == '-' || val.charAt(87) == '+') {  temp = Integer.*parseInt*(val.substring(87, 92));  }    //extract quality parameter at 93rd position of record.  String quality = val.substring(92, 93);    //check if temperature is not missing and quality code belongs to any one of 0,1,4,5,9.If yes then write year and temperature to context.  **if**(temp != ***MISSING*** && quality.matches("[01459]")) {  context.write(**new** Text(year), **new** IntWritable(temp));  }  }  } |
|  |
| **Code for Reducer:**  **public** **class** MaxTemperatureReducer **extends** Reducer<Text, IntWritable, Text, IntWritable> {  **public** **void** reduce(Text key, Iterable<IntWritable> values, Context context)  **throws** IOException, InterruptedException {  // process values  //declare integer variable for storing maximum value  **int** maxTemp = -9999;    //find maximum value of temperature in given year here  **for**(IntWritable val: values) {  **if**(val.get() > maxTemp) {  maxTemp = val.get();  }  }    //write key and value to the context  context.write(key, **new** IntWritable(maxTemp));  }  } |
|  |
| **Code for Driver Class:**  **public** **class** MaxTemperature {  **public** **static** **void** main(String[] args) **throws** Exception {    Configuration conf = **new** Configuration();  Job job = Job.*getInstance*(conf, "maxtemp");  job.setJarByClass(MaxTemperature.**class**);  // **TODO**: specify a mapper  job.setMapperClass(MaxTemperatureMapper.**class**);    // **TODO**: specify a reducer  job.setReducerClass(MaxTemperatureReducer.**class**);  //job.setNumReduceTasks(3);  // **TODO**: specify output types  job.setOutputKeyClass(Text.**class**);  job.setOutputValueClass(IntWritable.**class**);    // **TODO**: specify input and output DIRECTORIES (not files)  FileInputFormat.*setInputPaths*(job, **new** Path("hdfs://localhost:9000/input"));  FileOutputFormat.*setOutputPath*(job, **new** Path("hdfs://localhost:9000/output/maxTempResult"));    System.***out***.println(job.waitForCompletion(**true**));    }  } |
|  |
|  |

|  |
| --- |
| **P**C**o**an**st**w**La**e**b**u**:**se combiner function in the above code? If yes, write the above code with combiner function.  **Code for postlab question**  Yes we can use combiner in the code to find max temperature since, the logic for finding max temperature that we implemented is both associative (since we find maximum by going through data 1 at a time) and commutative (since we can find maximum regardless of the order of checking conditions in data)  Code: (Add to driver class)  job.setCombinerClass(MaxTemperatureReducer.**class**);  The combiner class must implement the reduce function and must have the same output class as Reducer. Here, the already implemented reducer is enough to perform the combine function. |