Q1 MapReduce

Here, we have chosen the stock market dataset on which we have performed map-reduce

operations. Following is the structure of the data. Kindlyfind the solutions to the questions

below.

Ans:

JavaCode ->

**import** java.io.\*;

**import** org.apache.hadoop.io.Text;

**import** org.apache.hadoop.io.LongWritable;

**import** org.apache.hadoop.io.DoubleWritable;

**import** org.apache.hadoop.mapreduce.Job;

**import** org.apache.hadoop.mapreduce.Mapper;

**import** org.apache.hadoop.mapreduce.Reducer;

**import** org.apache.hadoop.conf.\*;

**import** org.apache.hadoop.fs.\*;

**import** org.apache.hadoop.mapreduce.lib.input.\*;

**import** org.apache.hadoop.mapreduce.lib.output.\*;

**public** **class** ExamAllTimeHigh {

**public** **static** **class** MapClass **extends** Mapper<LongWritable,Text,Text,DoubleWritable>

{

**private** Text stock\_id = **new** Text();

**private** DoubleWritable High = **new** DoubleWritable();

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

{

**try**{

String[] str = value.toString().split(",");

**double** high = Double.*parseDouble*(str[4]);

stock\_id.set(str[1]);

High.set(high);

//context.write(new Text(str[1]),new LongWritable(vol));

context.write(stock\_id, High);

}

**catch**(Exception e)

{

System.***out***.println(e.getMessage());

}

}

}

**public** **static** **class** ReduceClass **extends** Reducer<Text,DoubleWritable,Text,DoubleWritable>

{

**private** DoubleWritable result = **new** DoubleWritable();

**public** **void** reduce(Text key, Iterable<DoubleWritable> values,Context context) **throws** IOException, InterruptedException {

**double** maxValue=0;

**double** temp\_val=0;

**for** (DoubleWritable value : values) {

temp\_val = value.get();

**if** (temp\_val > maxValue) {

maxValue = temp\_val;

}

}

result.set(maxValue);

context.write(key, result);

//context.write(key, new LongWritable(sum));

}

}

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

Configuration conf = **new** Configuration();

//conf.set("name", "value")

//conf.set("mapreduce.input.fileinputformat.split.minsize", "134217728");

Job job = Job.*getInstance*(conf, "Highest Price for each stock");

job.setJarByClass(ExamAllTimeHigh.**class**);

job.setMapperClass(MapClass.**class**);

//job.setCombinerClass(ReduceClass.class);

job.setReducerClass(ReduceClass.**class**);

job.setNumReduceTasks(1);

job.setOutputKeyClass(Text.**class**);

job.setOutputValueClass(DoubleWritable.**class**);

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

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

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

}

}

OP IN HUE ->

Graphical user interface, application, table

Description automatically generated

Hive

Write a program to find the count of customers for each profession.

Query ->

select profession,count(cust\_id) as total from customer group by profession order by total limit 10;

O/P ->

Social Worker 1

Writer 101

Artist 175

Environmental scientist 176

Carpenter 181

Dancer 185

Therapist 187

Economist 189

Real estate agent 191

Electrical engineer 192

ScreenShots (Querry)->

A picture containing text

Description automatically generated

ScreenShots-o/p>

Text

Description automatically generated

Q—2(2) Write a program to find the top 10 products sales wise

Qyerry->

select product,count(amount) as total from sales group by product order by total desc limit 15;

O/P =

Lawn Games 466

Swing Sets 464

Golf 452

Cardio Machine Accessories 445

Yoga & Pilates 444

Mahjong 437

Hockey 434

Boxing 431

Basketball 431

Weightlifting Belts 429

Cross-Country Skiing 429

Sandboxes 428

Skating 427

Balance Beams 425

Bodyboarding 425

ScreenShot(Querry) -> 

ScreenShot o/p->

Text

Description automatically generated

Q2(3)) Write a program to create partiioned table on category

Queryy->

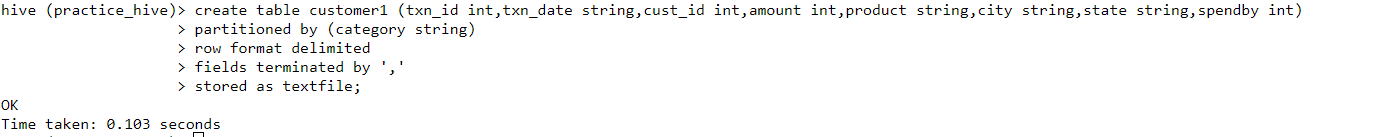
create table customer1 (txn\_id int,txn\_date string,cust\_id int,amount int,product string,city string,state string,spendby int)

> partitioned by (category string)

> row format delimited

> fields terminated by ','

> stored as textfile;



PySpark –

What was the highest number of people travelled in which year?

Ans

1. airlineRDD = sc.textFile("/user/bigdatamind4389/Exam1")
2. >>> airlineRDD1 = airlineRDD.map(lambda a:a.encode("ascii","ignore"))
3. >>> header = airlineRDD1.first()
4. >>> airlineRDD2 =airlineRDD1.filter(lambda a:a!=header)
5. >>> arrayRDD = airlineRDD2.map(lambda a:a.split(','))
6. >>> arrayRDD1=arrayRDD.map(lambda a:(a[0],int(a[3])))
7. >>> arrayRDD2 = arrayRDD1.reduceByKey(lambda a,b:a+b)
8. >>> sortbyval = arrayRDD2.sortBy(lambda a:-a[1])
9. >>> for i in sortbyval.take(1):
10. ... print(i)
11. ... ('2007', 176299)

O/p->

('2007', 176299)

Screenshot(Program with o/p)->

Text

Description automatically generated

Q2) Identifying the highest revenue generation for which year

Program->

airlineRDD = sc.textFile("/user/bigdatamind4389/Exam1")

>>> airlineRDD1 = airlineRDD.map(lambda a:a.encode("ascii","ignore"))

>>> header = airlineRDD1.first()

>>> airlineRDD2 =airlineRDD1.filter(lambda a:a!=header)

>>> arrayRDD = airlineRDD2.map(lambda a:a.split(','))

>>> arrayRDD1=arrayRDD.map(lambda a:(a[0],float(a[2])))

>>> arrayRDD2 = arrayRDD1.reduceByKey(lambda a,b:a+b)

>>> sortbyval = arrayRDD2.sortBy(lambda a:-a[1])

>>> for i in sortbyval.take(1):

... print(i)

... ('2014', 1566.8)

OP== ('2014', 1566.8)

ScreenShot(with o/p)->

Text, letter

Description automatically generated

Q3) Identifying the highest revenue generation for which year and quarter (Common

group)

airlineRDD = sc.textFile("/user/bigdatamind4389/Exam1")

>>> airlineRDD1 = airlineRDD.map(lambda a:a.encode("ascii","ignore"))

>>> header = airlineRDD1.first()

>>> airlineRDD2 =airlineRDD1.filter(lambda a:a!=header)

>>> arrayRDD = airlineRDD2.map(lambda a:a.split(','))

>>> key\_val = arrayRDD.map(lambda a:(a[0]+" "+a[1],float(a[2])))

>>> arrayRDD2 = key\_val.reduceByKey(lambda a,b:a+b)

>>> sortbyval = arrayRDD2.sortBy(lambda a:-a[1])

>>> for i in sortbyval.take(1):

... print(i)

...

('2014 3', 396.37)

O/P = ('2014 3', 396.37)

Text

Description automatically generated