PIG SOLUTION

----------------------SOLUTION 1 -------------------------------------**METHOD 1**

grunt> TitanicData = LOAD '/home/acadgild/Desktop/Abhinav/TitanicData.txt' USING PigStorage(',') AS (pid,sur,class,name,sex,age,sib,parch,tkt,fare,cabin,embark);

grunt> ClassFare = FOREACH TitanicData GENERATE class,fare;

grunt> GroupClassFare = GROUP ClassFare By class;

grunt> AvgClassFare = FOREACH GroupClassFare GENERATE ClassFare.class, AVG(ClassFare.fare);

({(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1)},84.1546874999999)

({(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2)},20.662183152173906)

({(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3)},13.67555010183301)

**METHOD 2**

grunt> GroupClassFare = GROUP TitanicData By class;

grunt> AvgClassFare = FOREACH GroupClassFare GENERATE TitanicData.class, AVG(TitanicData.fare);

({(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1),(1)},84.1546874999999)

({(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2),(2)},20.662183152173906)

({(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3),(3)},13.67555010183301)

----------------------SOLUTION 2 -------------------------------------

grunt> FilterAlive = FILTER TitanicData BY sur==0;

grunt> GroupAlive= GROUP FilterAlive BY class;

grunt> CountAlive = FOREACH GroupAlive GENERATE group, COUNT(FilterAlive.name);

(1,80)

(2,97)

(3,372)l

grunt> FilterAliveS = FILTER TitanicData BY (sur==0) AND (embark=='S');

grunt> GroupAliveS = GROUP FilterAliveS BY class;

grunt> CountAliveS = FOREACH GroupAliveS GENERATE group, COUNT(FilterAliveS.name);

(1,53)

(2,88)

(3,286)

----------------------SOLUTION 3 -------------------------------------

grunt> FilterDeadMale = FILTER TitanicData BY (sur==1) AND (sex=='male');

grunt> GroupDeadMale= GROUP FilterDeadMale BY class;

grunt> CountDeadMale = FOREACH GroupDeadMale GENERATE group, COUNT(FilterDeadMale.name);

(1,45)

(2,17)

(3,47)

grunt> FilterDeadFemale = FILTER TitanicData BY (sur==1) AND (sex=='female');

grunt> GroupDeadFemale = GROUP FilterDeadFemale BY class;

grunt> CountDeadFemale = FOREACH GroupDeadFemale GENERATE group, COUNT(FilterDeadFemale.name);

(1,91)

(2,70)

(3,72)

----------------------------------------------------------------------

MAP REDUCE SOLUTION

**------------------Problem 1 Solution---------------------------------------------------------------MAPPER-----------------------------------**package Titanic;

import java.io.IOException;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

public class TitanicP1Mapper extends Mapper<LongWritable,Text,Text,Text> {

@Override

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

throws IOException, InterruptedException{

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

if(values[9]!= null && !values[9].isEmpty())

{

context.write(new Text(values[2]), new Text(values[9]));

}

}

}

**-----------------------------REDUCER----------------------------------**

package Titanic;

import java.io.IOException;

import org.apache.hadoop.mapreduce.Reducer;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.io.DoubleWritable;

public class TitanicP2Reducer extends Reducer<Text,Text,Text,DoubleWritable>{

@Override

public void reduce(Text key, Iterable<Text> value, Context context)

throws IOException, InterruptedException{

int count=0;

double sum=0;

double avgFare=0;

for(Text values : value){

sum=sum+Double.parseDouble(values.toString());

count=count+1;

}

avgFare=sum/count;

context.write(key, new DoubleWritable(avgFare));

}

}

**-----------------------------DRIVER-----------------------------------**

package Titanic;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.conf.Configured;

import org.apache.hadoop.util.Tool;

import org.apache.hadoop.util.ToolRunner;

import org.apache.hadoop.io.DoubleWritable;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

import org.apache.hadoop.fs.Path;

public class TitanicP1Driver extends Configured implements Tool{

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

ToolRunner.run(new Configuration(), new TitanicP1Driver(), abhi);

}

@Override

public int run(String[] abhi) throws Exception{

//Configuration conf = new Configuration();

Job job = Job.getInstance();

job.setJobName("Abhinav Titanic Prob 1");

job.setJarByClass(TitanicP1Driver.class);

job.setMapperClass(TitanicP1Mapper.class);

job.setReducerClass(TitanicP2Reducer.class);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(Text.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(DoubleWritable.class);

job.setInputFormatClass(TextInputFormat.class);

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

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

job.waitForCompletion(true);

return 0;

}

}

**-----------------------------OUTPUT-----------------------------------**

[acadgild@localhost Desktop]$ hadoop fs -cat /Abhinav/TitanicP1output1/part-r-00000

16/05/31 22:29:43 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

1 84.1546874999999

2 20.662183152173906

3 13.67555010183301

[acadgild@localhost Desktop]$

**------------------Problem 2 Solution----------------------------------**

**-----------------------------MAPPER-----------------------------------**

package Titanic;

import java.io.IOException;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.io.IntWritable;

public class TitanicP2Mapper extends Mapper<LongWritable,Text,Text,IntWritable>{

@Override

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

throws IOException, InterruptedException{

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

if(values[1].equalsIgnoreCase("0") && values[11].equalsIgnoreCase("S")){

context.write(new Text(values[2]), new IntWritable(1));

}

}

}

**-----------------------------REDUCER----------------------------------**

package Titanic;

import java.io.IOException;

import org.apache.hadoop.mapreduce.Reducer;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.io.IntWritable;

public class TitanicP2bReducer extends Reducer<Text,IntWritable, Text, IntWritable>{

@Override

public void reduce(Text key, Iterable<IntWritable> value,Context context)

throws IOException, InterruptedException{

int count=0;

for(IntWritable values : value){

count=count+values.get();

}

context.write(new Text(key), new IntWritable(count));

}

}

**-----------------------------DRIVER-----------------------------------**package Titanic;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.conf.Configured;

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.util.Tool;

import org.apache.hadoop.util.ToolRunner;

import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class TitanicP2Driver extends Configured implements Tool{

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

ToolRunner.run(new Configuration(), new TitanicP2Driver(), neeti);

}

@Override

public int run(String[] neeti) throws Exception {

Job job= Job.getInstance();

job.setJobName("Neeti Titanic Program 2");

job.setJarByClass(TitanicP2Driver.class);

job.setMapperClass(TitanicP2Mapper.class);

job.setReducerClass(TitanicP2bReducer.class);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(IntWritable.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);

job.setInputFormatClass(TextInputFormat.class);

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

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

job.waitForCompletion(true);

return 0;

}

}

**-----------------------------OUTPUT-----------------------------------**

[acadgild@localhost Desktop]$ hadoop fs -cat /Abhinav/TitanicP1output2/part-r-00000

16/05/31 23:49:33 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

1 53

2 88

3 286

[acadgild@localhost Desktop]$

------------------Problem 3 Solution----------------------------------

**-----------------------------MAPPER-----------------------------------**

package Titanic;

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 TitanicP3Mapper extends Mapper<LongWritable,Text,Text,IntWritable> {

@Override

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

throws IOException, InterruptedException{

int count =1;

String classsex;

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

if(values[1].equalsIgnoreCase("1"))

{

//CompositeGroupKey classsex = new CompositeGroupKey(values[2], values[4]);

classsex = values[2]+"\t"+values[4];

context.write(new Text(classsex), new IntWritable(count));

}

}

}

**-----------------------------REDUCER----------------------------------**

package Titanic;

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Reducer;

public class TitanicP3Reducer extends Reducer<Text,IntWritable,Text,IntWritable>{

@Override

public void reduce(Text key, Iterable<IntWritable> value, Context context)

throws IOException, InterruptedException{

int sum=0;

for(IntWritable values : value){

sum=sum+values.get();

}

context.write(key, new IntWritable(sum));

}

}

**-----------------------------DRIVER-----------------------------------**

package Titanic;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.conf.Configured;

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.input.TextInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

import org.apache.hadoop.util.Tool;

import org.apache.hadoop.util.ToolRunner;

public class TitanicP3Driver extends Configured implements Tool{

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

ToolRunner.run(new Configuration(), new TitanicP3Driver(), abhi);

}

@Override

public int run(String[] abhi) throws Exception {

Job job= Job.getInstance();

job.setJobName("Abhi Titanic Program Problem 3");

job.setJarByClass(TitanicP3Driver.class);

job.setMapperClass(TitanicP3Mapper.class);

job.setReducerClass(TitanicP3Reducer.class);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(IntWritable.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);

job.setInputFormatClass(TextInputFormat.class);

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

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

job.waitForCompletion(true);

return 0;

}

}

----------------------------------------------------------------------

HIVE SOLUTION

hive> create table Titanic(pid int, sur int, class int, name string, sex string, age int, sib int, parch int, tkt string, fare int, cabin int, embark string) row format delimited fields terminated by ',' stored as textfile;

hive> load data local inpath '/home/acadgild/Desktop/Abhinav/TitanicData.txt' into table Titanic;

hive> select class, avg(fare) from titanic group by class ;

1 83.74537037037037

2 20.456521739130434

3 13.173116089613035

hive> select class, count(\*) from titanic where sur=0 GROUP BY class;

1 80

2 97

3 372

hive> select class, count(\*) from titanic where sur=0 and embark ='S' GROUP BY class;

1 53

2 88

3 286

hive> select class, sex, count(\*) from titanic where sur=1 group by class, sex order by sex, class;

1 female 91

2 female 70

3 female 72

1 male 45

2 male 17

3 male 47

----------------------------------------------------------------------