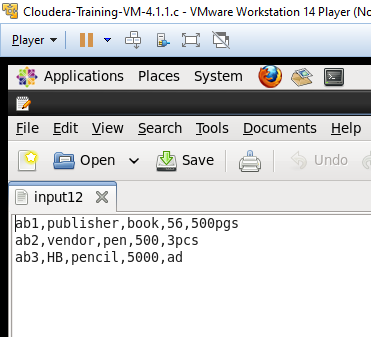
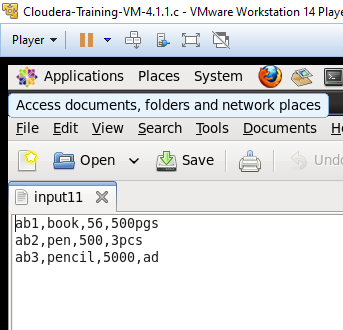
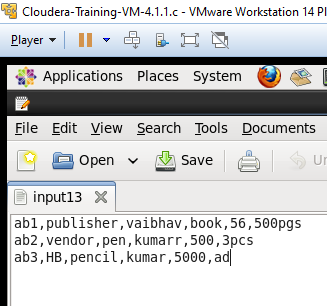
**Multiple File Input**

**Problem Statement:** Create 2 or 3 input files on your own , in which the data is present in different format. Write a program to process the these files using different map class and perform any one aggerate function like sum, max, min etc. on it.





CODE:

import java.io.IOException;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

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

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

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

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Reducer;

import org.apache.hadoop.util.GenericOptionsParser;

public class MultipleFileInp {

public static class map1 extends Mapper <LongWritable, Text, Text,IntWritable>

{

@Override

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

throws IOException, InterruptedException{

String line = value.toString();

String[] line1=line.split(",");

String category=line1[0];

Text outputKey =new Text(category);

int salary=Integer.parseInt(line1[2]);

IntWritable outputValue=new IntWritable(salary);

con.write(outputKey, outputValue);

}

}

public static class map2 extends Mapper <LongWritable, Text, Text,IntWritable>

{

@Override

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

throws IOException, InterruptedException{

String line = value.toString();

String[] line1=line.split(",");

String category=line1[0];

Text outputKey =new Text(category);

int salary=Integer.parseInt(line1[3]);

IntWritable outputValue=new IntWritable(salary);

con.write(outputKey, outputValue);

}

}

public static class map3 extends Mapper <LongWritable, Text, Text,IntWritable>

{

@Override

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

throws IOException, InterruptedException{

String line = value.toString();

String[] line1=line.split(",");

String category=line1[0];

Text outputKey =new Text(category);

int salary=Integer.parseInt(line1[4]);

IntWritable outputValue=new IntWritable(salary);

con.write(outputKey, outputValue);

}

}

public static class red extends Reducer<Text,IntWritable,Text,IntWritable>{

public void reduce(Text category,Iterable<IntWritable> total\_sal,Context con)throws IOException,InterruptedException{

int sum =0;

for(IntWritable value:total\_sal){

sum+=value.get();

}

con.write(category, new IntWritable(sum));

}

}

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

Configuration c = new Configuration();

String[]files = new GenericOptionsParser(c,args).getRemainingArgs();

Path p1= new Path(files[0]);

Path p2= new Path(files[1]);

Path p3= new Path(files[2]);

Path p4= new Path(files[3]);

Job j =new Job(c,"multiple");

j.setJarByClass(MultipleFileInp.class);

j.setMapperClass(map1.class);

j.setMapperClass(map2.class);

j.setMapperClass(map3.class);

j.setReducerClass(red.class);

j.setMapOutputKeyClass(Text.class);

j.setMapOutputValueClass(IntWritable.class);

MultipleInputs.addInputPath(j, p1, TextInputFormat.class, map1.class);

MultipleInputs.addInputPath(j, p2, TextInputFormat.class, map2.class);

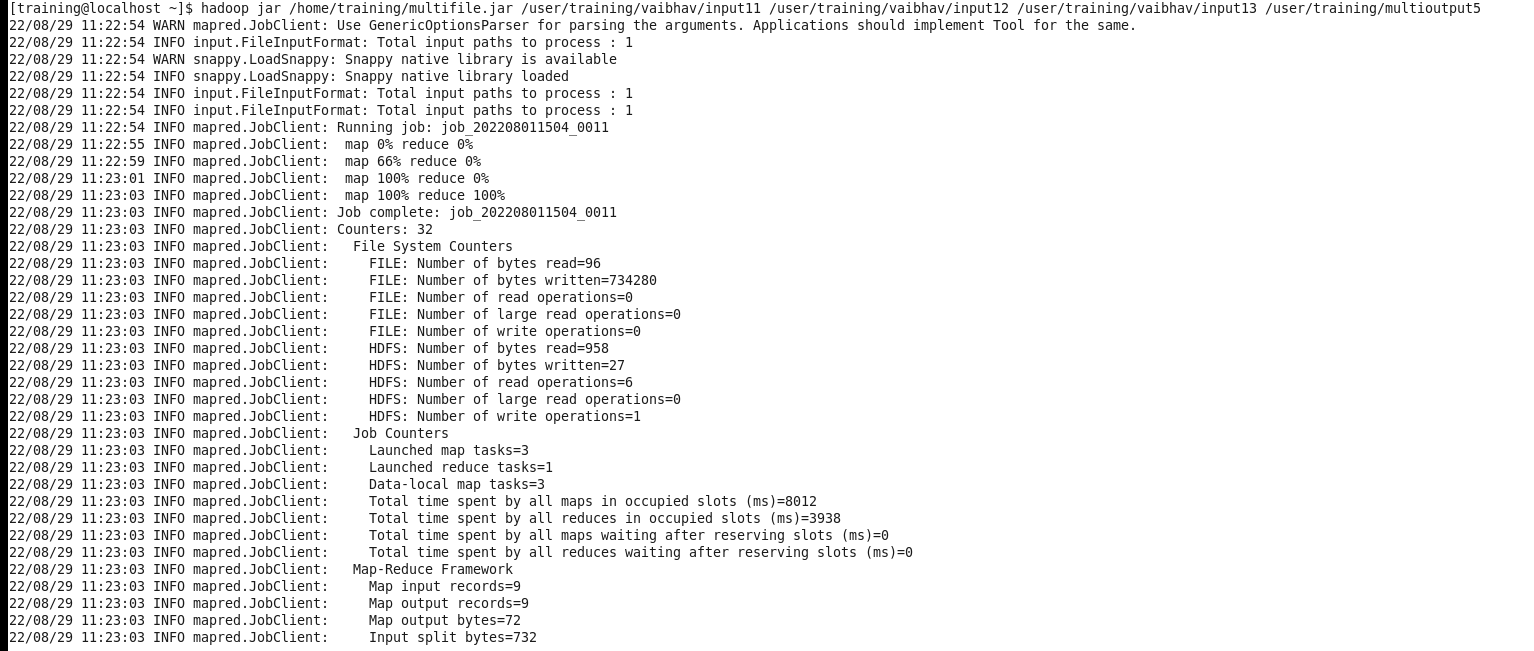
MultipleInputs.addInputPath(j, p3, TextInputFormat.class, map3.class);

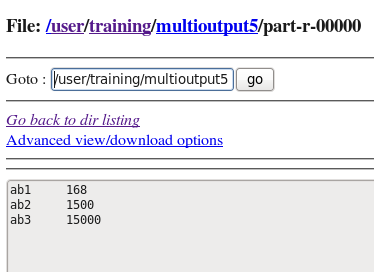
FileOutputFormat.setOutputPath(j, p4);

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

}

}





**MAX**

import java.io.IOException;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Reducer;

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

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

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

import org.apache.hadoop.util.GenericOptionsParser;

public class MultipleFileInp {

public static class map1 extends Mapper<LongWritable,Text,Text,IntWritable>

{

public void map(LongWritable key, Text value, Context con) throws IOException, InterruptedException

{

String line = value.toString();

String[] line1 = line.split(",");

String name = line1[0];

Text outputKey = new Text(name);

int cost = Integer.parseInt(line1[2]);

IntWritable outputValue = new IntWritable(cost);

con.write(outputKey, outputValue);

} }

public static class Map2 extends Mapper<LongWritable, Text, Text, IntWritable>

{

public void map(LongWritable key, Text value, Context con) throws IOException, InterruptedException

{

String line = value.toString();

String[] line1 = line.split(",");

String name = line1[0];

Text outputKey = new Text(name);

int cost = Integer.parseInt(line1[3]);

IntWritable outputValue = new IntWritable(cost);

con.write(outputKey, outputValue);

} }

public static class Map3 extends Mapper<LongWritable, Text, Text, IntWritable>

{

public void map(LongWritable key, Text value, Context con) throws IOException, InterruptedException

{

String line = value.toString();

String[] line1 = line.split(",");

String name = line1[0];

Text outputKey = new Text(name);

int cost = Integer.parseInt(line1[4]);

IntWritable outputValue = new IntWritable(cost);

con.write(outputKey, outputValue);

}

}

public static class Red extends Reducer<Text, IntWritable, Text,IntWritable>

{

public void reduce(Text name, Iterable<IntWritable>total\_cost, Context con) throws IOException, InterruptedException

{

**int sum = 0;**

**int max = 0;**

**for (IntWritable value : total\_cost) {**

**if (value.get() > max) {**

**max = value.get();**

**}**

**}**

con.write(name,new IntWritable(sum));

}

}

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

Configuration c = new Configuration();

GenericOptionsParser parser = new GenericOptionsParser(c,args);

String[] files = parser.getRemainingArgs();

Path p1 = new Path(files[0]);

Path p2 = new Path(files[1]);

Path p3 = new Path(files[2]);

Path p4 = new Path(files[3]);

Job j = new Job(c,"multiple");

j.setJarByClass(MultipleFileInp .class);

j.setMapperClass(map1.class);

j.setMapperClass(Map2.class);

j.setMapperClass(Map3.class);

j.setReducerClass(Red.class);

j.setOutputKeyClass(Text.class);

j.setOutputValueClass(IntWritable.class);

MultipleInputs.addInputPath(j,p1,TextInputFormat.class,map1.class);

MultipleInputs.addInputPath(j, p2, TextInputFormat.class,Map2.class);

MultipleInputs.addInputPath(j, p3, TextInputFormat.class,Map3.class);

FileOutputFormat.setOutputPath(j,p4);

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



