**MapReduce Project**

**1. wordcount - Counting the frequency of Teams.**

**ClsMapper.java**

package wordcount;

import java.util.StringTokenizer;

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 ClsMapper extends Mapper<LongWritable, Text, Text, IntWritable>

{

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

{ try

{ StringTokenizer token = new StringTokenizer(value.toString());

while(token.hasMoreTokens())

{ context.write(new Text(token.nextToken()), new IntWritable(1)); }

}

catch(Exception e)

{ System.out.println(e.getMessage()); }

}

}

**ClsReducer.java**

package wordcount;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Reducer;

public class ClsReducer extends Reducer<Text, IntWritable, Text, LongWritable>

{

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

{ try

{ Long sum = 0L;

for(IntWritable value : values)

{ sum += value.get(); }

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

}

catch(Exception e)

{ System.out.println(e.getMessage()); }

}

}

**ClsDriver.java**

package wordcount;

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.LongWritable;

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 ClsDriver extends Configured implements Tool

{ @Override

public int run(String[] arg0) throws Exception

{

Job job = Job.getInstance();

job.setJobName("Word Count");

job.setJarByClass(getClass());

job.setMapperClass(ClsMapper.class);

job.setReducerClass(ClsReducer.class);

job.setInputFormatClass(TextInputFormat.class);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(IntWritable.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(LongWritable.class);

FileInputFormat.setInputPaths(job, new Path(arg0[0]));

FileOutputFormat.setOutputPath(job, new Path(arg0[1]));

return job.waitForCompletion(true)?1:0;

}

public static void main(String[] args)

{ try

{ ToolRunner.run(new Configuration(), new ClsDriver(), args); }

catch(Exception e)

{ System.out.println(e.getMessage()); }

}

}

**ipl.txt**

KIXP MI DD SRH KKR RR RCB

KIXP MI DD SRH KKR RR RCB DD SRH KKR RR

KIXP SRH KKR RR RCB

KIXP MI DD SRH KKR RR RCB

KIXP MI DD RR RCB

KIXP MI DD SRH KKR RR DD SRH KKR RR

KIXP MI DD KKR RR RCB

KIXP MI DD SRH KKR RR RCB DD SRH KKR RR

KIXP MI DD SRH KKR RR RCB

MI DD SRH KKR RR RCB

KIXP MI DD SRH KKR RR RCB

KIXP MI DD SRH KKR RR RCB

**Output**

DD 15

KIXP 12

KKR 15

MI 14

RCB 12

RR 16

SRH 14

**2. nas - Finding the maximum scores of both Genders.**

**ClsMapper.java**

package nas;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Mapper;

public class ClsMapper extends Mapper<Text, Text, Text, IntWritable>

{

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

{ try

{

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

context.write(key, new IntWritable(Integer.parseInt(values[2])));

}

catch(Exception e)

{ System.out.println(e.getMessage()); }

}

}

**ClsReducer.java**

package nas;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Reducer;

public class ClsReducer extends Reducer<Text, IntWritable, Text, IntWritable>

{

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

{ try

**nameAgeScore.txt**

f,alia,23,87

m,ali,27,82

m,alice,24,57

f,allen,23,87

f,alok,22,37

f,john,21,89

m,jason,24,57

f,drakula,26,47

f,erik,24,92

{

Integer max = Integer.MIN\_VALUE;

for(IntWritable value : values)

{ if(value.get() > max)

{ max = value.get(); }

}

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

}

catch(Exception e)

{ System.out.println(e.getMessage()); }

}

}

**ClsDriver.java**

**Output**

f 92

m 82

package nas;

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.KeyValueTextInputFormat;

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

import org.apache.hadoop.util.Tool;

import org.apache.hadoop.util.ToolRunner;

public class ClsDriver extends Configured implements Tool

{ @Override

public int run(String[] arg0) throws Exception

{

Configuration conf = new Configuration();

conf.set("mapreduce.input.keyvaluelinerecordreader.key.value.separator",",");

Job job = Job.getInstance(conf);

job.setJobName("Name Age Score");

job.setJarByClass(getClass());

job.setMapperClass(ClsMapper.class);

job.setReducerClass(ClsReducer.class);

job.setInputFormatClass(KeyValueTextInputFormat.class);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(IntWritable.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);

FileInputFormat.setInputPaths(job, new Path(arg0[0]));

FileOutputFormat.setOutputPath(job, new Path(arg0[1]));

return job.waitForCompletion(true)?1:0;

}

public static void main(String[] args)

{ try

{ ToolRunner.run(new Configuration(), new ClsDriver(), args); }

catch(Exception e)

{ System.out.println(e.getMessage()); }

}

}

**3. wordcountCombiner - Counting the frequency of Teams.**

**ClsMapper.java**

package wordcountCombiner;

import java.util.StringTokenizer;

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 ClsMapper extends Mapper<LongWritable, Text, Text, IntWritable>

{

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

{ try

{

StringTokenizer token = new StringTokenizer(value.toString());

while(token.hasMoreTokens())

{ context.write(new Text(token.nextToken()), new IntWritable(1)); }

}

catch(Exception e)

{ System.out.println(e.getMessage()); }

}

}

**ClsReducer.java**

package wordcountCombiner;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Reducer;

public class ClsReducer extends Reducer<Text, IntWritable, Text, LongWritable>

{

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

{ try

{

Long sum = 0L;

for(IntWritable value : values)

{ sum += value.get(); }

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

}

catch(Exception e)

{ System.out.println(e.getMessage()); }

}

}

**ClsDriver.java**

package wordcountCombiner;

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.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Job;

**Output**

DD 612863

KIXP 677373

KKR 645119

MI 645119

RCB 612861

RR 741887

SRH 612863

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 ClsDriver extends Configured implements Tool

{ @Override

public int run(String[] arg0) throws Exception

{

Job job = Job.getInstance();

job.setJobName("Word Count");

job.setJarByClass(getClass());

job.setMapperClass(ClsMapper.class);

job.setReducerClass(ClsReducer.class);

job.setCombinerClass(ClsCombiner.class);

job.setInputFormatClass(TextInputFormat.class);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(IntWritable.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(LongWritable.class);

FileInputFormat.setInputPaths(job, new Path(arg0[0]));

FileOutputFormat.setOutputPath(job, new Path(arg0[1]));

return job.waitForCompletion(true)?1:0;

}

public static void main(String[] args)

{ try

{ ToolRunner.run(new Configuration(), new ClsDriver(), args); }

catch(Exception e)

{ System.out.println(e.getMessage()); }

}

}

**ClsCombiner.java**

package wordcountCombiner;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Reducer;

public class ClsCombiner extends Reducer<Text, IntWritable, Text, IntWritable>

{

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

{ try

{ Integer sum = 0;

for(IntWritable value : values)

{ sum += value.get(); }

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

}

catch(Exception e)

{ System.out.println(e.getMessage()); }

}

}

**ipl1.txt is same as ipl.txt but it has very large size (There are so many records i.e. repeated).**

**4. nasCP - Partitioning according to the Age (<25, 25-35, >35) of both genders.**

**ClsMapper.java**

package nasCP;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Mapper;

public class ClsMapper extends Mapper<Text, Text, Text, DTNameAgeScore>

{

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

{ try

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

DTNameAgeScore out = new DTNameAgeScore();

out.name.set(values[0]);

out.age.set(Integer.parseInt(values[1]));

out.score.set(Integer.parseInt(values[2]));

context.write(key, out);

}

catch(Exception e)

{ System.out.println(e.getMessage()); }

}

}

**ClsReducer.java**

package nasCP;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Reducer;

public class ClsReducer extends Reducer<Text, DTNameAgeScore, Text, DTNameAgeScore>

{

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

{ try

{ Integer max = Integer.MIN\_VALUE;

DTNameAgeScore out = new DTNameAgeScore();

for(DTNameAgeScore value : values)

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

{ max = value.score.get();

out.name.set(value.name);

out.age.set(value.age.get());

out.score.set(value.score.get());

}

}

context.write(key, out);

}

catch(Exception e)

{ System.out.println(e.getMessage()); }

}

}

**ClsDriver.java**

package nasCP;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.conf.Configured;

import org.apache.hadoop.fs.Path;

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.KeyValueTextInputFormat;

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

import org.apache.hadoop.util.Tool;

import org.apache.hadoop.util.ToolRunner;

public class ClsDriver extends Configured implements Tool

{ @Override

public int run(String[] arg0) throws Exception

{

Configuration conf = new Configuration();

conf.set("mapreduce.input.keyvaluelinerecordreader.key.value.separator",",");

Job job = Job.getInstance(conf);

job.setJobName("Name Age Score");

job.setJarByClass(getClass());

job.setNumReduceTasks(3);

job.setMapperClass(ClsMapper.class);

job.setReducerClass(ClsReducer.class);

job.setPartitionerClass(ClsPartitioner.class);

job.setInputFormatClass(KeyValueTextInputFormat.class);

job.setMapOutputKeyClass(Text.class);

job.setMapOutputValueClass(DTNameAgeScore.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(DTNameAgeScore.class);

FileInputFormat.setInputPaths(job, new Path(arg0[0]));

FileOutputFormat.setOutputPath(job, new Path(arg0[1]));

return job.waitForCompletion(true)?1:0;

}

public static void main(String[] args)

{ try

{ ToolRunner.run(new Configuration(), new ClsDriver(), args); }

catch(Exception e)

{ System.out.println(e.getMessage()); }

}

}

**ClsPartitioner.java**

package nasCP;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Partitioner;

public class ClsPartitioner extends Partitioner<Text, DTNameAgeScore>

{ @Override

public int getPartition(Text key, DTNameAgeScore value, int numPartition)

{

if(value.age.get() <= 25)

return 3%numPartition;

else if(value.age.get() > 25 && value.age.get() <= 35)

return 2%numPartition;

else

return 1%numPartition;

}

}

**DTNameAgeScore.java**

package nasCP;

import java.io.DataInput;

import java.io.DataOutput;

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.io.Writable;

public class DTNameAgeScore implements Writable

{

Text name ;

IntWritable age,score;

public DTNameAgeScore()

{ name = new Text();

age = new IntWritable();

score = new IntWritable();

}

@Override

public void readFields(DataInput in) throws IOException

{ name.readFields(in);

age.readFields(in);

score.readFields(in);

}

@Override

public void write(DataOutput out) throws IOException

{ name.write(out);

age.write(out);

score.write(out);

}

public String toString()

{ return name+","+age+","+score; }

}

**nameAgeScore1.txt**

f,alia,30,87

m,ali,29,82

m,alice,24,57

f,allen,23,87

f,alok,22,37

f,john,21,89

m,jason,28,57

f,drakula,26,47

f,erik,24,92

f,alia,23,87

m,ali,31,82

m,alice,36,57

f,allen,43,87

f,alok,22,37

f,john,42,89

m,jason,37,57

f,drakula,36,47

f,erik,22,92

**Output**

**part-r-00000**

f erik,22,92

m alice,24,57

**part-r-00001**

f john,42,89

m jason,37,57

**part-r-00002**

f alia,30,87

m ali,31,82