Name : Chokhariya Nikita Anilkumar

Stream : MSCIT

semester : 3

Subject : Hadoop practical

Enrollnment no : 202300819010009

Program 1 :

```
import java.io.IOException;
import java.util.StringTokenizer;
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.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

public class FirstProgram {
      public static class Map extends
Mapper<LongWritable,Text,Text,IntWritable>{
            public void map(LongWritable key,Text value,Context context)
throws IOException,InterruptedException{
                  String line = value.toString();
                  StringTokenizer token = new StringTokenizer(line);
                  while(token.hasMoreElements()) {
                        value.set(token.nextToken());
                        context.write(value, new IntWritable(1));
                  }
            }
      }

      public static class Reduce extends
Reducer<Text,IntWritable,Text,IntWritable>{
            public void reduce(Text key,Iterable<IntWritable>
value,Context context) throws IOException,InterruptedException{
                  int sum = 0;
                  for(IntWritable i : value) {
                        sum += i.get();
                  }
                  context.write(key, new IntWritable(sum));
            }
      }

      public static void main(String args[]) throws Exception{
            Configuration conf = new Configuration();
            Job job = Job.getInstance(conf,"FirstProgram");
            job.setJarByClass(FirstProgram.class);
```

```
            job.setMapperClass(Map.class);
            job.setReducerClass(Reduce.class);

            job.setMapOutputKeyClass(Text.class);
            job.setMapOutputValueClass(IntWritable.class);

            job.setOutputKeyClass(Text.class);
            job.setOutputValueClass(IntWritable.class);

            job.setInputFormatClass(TextInputFormat.class);
            job.setOutputFormatClass(TextOutputFormat.class);

            Path outputPath = new Path(args[1]);
            FileInputFormat.addInputPath(job, new Path(args[0]));
            FileOutputFormat.setOutputPath(job, new Path(args[1]));

            outputPath.getFileSystem(conf).delete(outputPath, true);
            System.exit(job.waitForCompletion(true)?0:1);
      }
}
```

```
Text File :
Hadoop
Hadoop
Hadoop
Hadoop
Hadoop
Hadoop
Hadoop
HDFS
HDFS
HDFS
HDFS
HDFS
HDFS
HDFS
HDFS
HDFS
HDFS
```

```
Commands :
Put file in Hadoop file system :
hdfs dfs -put source destination
hadoop jar jar-path text-file-path-or-csv-file-path output-path
hdfs dfs -cat output-path/part-r-00000

Output :
```

```
Program 2 :
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.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class SecondProgram {
      public static class Map extends
Mapper<LongWritable,Text,Text,IntWritable>{
            public void map(LongWritable key,Text value,Context context)
throws IOException,InterruptedException{
                  String[] cols = value.toString().split(",");
                  String year = cols[0];
                  int temperature = Integer.parseInt(cols[1]);
                  context.write(new Text(year),new
IntWritable(temperature));
            }
      }

      public static class Reduce extends
Reducer<Text,IntWritable,Text,IntWritable>{
            public void reduce(Text key,Iterable<IntWritable>
values,Context context) throws IOException,InterruptedException{
                  int minTemp = Integer.MAX_VALUE;
                  for(IntWritable value : values) {
                        minTemp = Math.min(minTemp, value.get());
                  }
                  context.write(key, new IntWritable(minTemp));
            }
      }

      public static void main(String args[]) throws Exception {
            Configuration conf = new Configuration();
            Job job = Job.getInstance(conf,"SecondProgram");

            job.setJarByClass(SecondProgram.class);
            job.setMapperClass(Map.class);
            job.setReducerClass(Reduce.class);

            job.setOutputKeyClass(Text.class);
            job.setOutputValueClass(IntWritable.class);

            FileInputFormat.addInputPath(job,new Path(args[0]));
            FileOutputFormat.setOutputPath(job, new Path(args[1]));

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

Text File :
2014 1
2014 3
2014 -1
2014 5
2014 6
2014 8
2014 9
2014 10
2015 1
```

```
2015 -2
2015 5
2015 3
2015 4
```

Commands :

```
Put file in Hadoop file system :
hdfs dfs -put source destination
hadoop jar jar-path text-file-path-or-csv-file-path output-path
hdfs dfs -cat output-path/part-r-00000
```

Output :

Program 3 :

```
import java.io.IOException;
import java.util.StringTokenizer;
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.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class ThirdProgram {
      public static class Map extends
Mapper<LongWritable,Text,Text,IntWritable>{
            public void map(LongWritable key,Text value,Context context)
throws IOException,InterruptedException{
                  String line = value.toString();
                  StringTokenizer token = new StringTokenizer(line);
                  while(token.hasMoreElements()) {
                        value.set(token.nextToken());
                        context.write(value, new IntWritable(1));
                  }
            }
      }

      public static class Reduce extends
Reducer<Text,IntWritable,Text,IntWritable>{
            private int outerSum = 0;

            public void reduce(Text key,Iterable<IntWritable>
values,Context context) throws IOException,InterruptedException{
                  int sum = 0;
                  for(IntWritable value : values) {
                        sum += value.get();
                        outerSum += value.get();
```

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

          public void cleanup(Context context) throws
IOException,InterruptedException{
                int avg = outerSum / 2;
                context.write(new Text("Average"), new
IntWritable(avg));
          }
     }

     public static void main(String args[]) throws Exception {
          Configuration conf = new Configuration();
          Job job = Job.getInstance(conf,"ThirdProgram");

          job.setJarByClass(ThirdProgram.class);
          job.setMapperClass(Map.class);
          job.setReducerClass(Reduce.class);

          job.setOutputKeyClass(Text.class);
          job.setOutputValueClass(IntWritable.class);

          FileInputFormat.addInputPath(job, new Path(args[0]));
          FileOutputFormat.setOutputPath(job, new Path(args[1]));

          System.exit(job.waitForCompletion(true)?0:1);
     }
}
Text File :
Hadoop
Hadoop
Hadoop
Hadoop
Hadoop
Hotspot
Hotspot
Hotspot
Hotspot
Commands :
Put file in Hadoop file system :
hdfs dfs -put source destination
hadoop jar jar-path text-file-path-or-csv-file-path output-path
hdfs dfs -cat output-path/part-r-00000


Output :
```

Program 4 :

```
import java.io.IOException;
import java.util.StringTokenizer;
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.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class FourthProgram {
      public static class Map extends
Mapper<LongWritable,Text,Text,IntWritable>{
            public void map(LongWritable key,Text value,Context context)
throws IOException,InterruptedException{
                  String line = value.toString();
                  StringTokenizer token = new StringTokenizer(line);
                  while(token.hasMoreElements()) {
                        value.set(token.nextToken());
                        if(value.getLength() >= 4) {
                              context.write(value, new IntWritable(1));
                        }
                  }
            }
      }

      public static class Reduce extends
Reducer<Text,IntWritable,Text,IntWritable>{
            private int cnt = 0;
            public void reduce(Text key,Iterable<IntWritable>
values,Context context) throws IOException,InterruptedException{
                  for(IntWritable value : values) {
                        cnt += value.get();
                  }
            }

            public void cleanup(Context context) throws
IOException,InterruptedException{
                  context.write(new Text("no of Count : "), new
IntWritable(cnt));
            }
      }

      public static void main(String args[]) throws Exception {
            Configuration conf = new Configuration();
            Job job = Job.getInstance(conf,"FourthProgram");

            job.setJarByClass(FourthProgram.class);
            job.setMapperClass(Map.class);
            job.setReducerClass(Reduce.class);

            job.setOutputKeyClass(Text.class);
            job.setOutputValueClass(IntWritable.class);

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

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

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

```
Text File :
Java
Python
C
C++

Commands :
Put file in Hadoop file system :
hdfs dfs -put source destination
hadoop jar jar-path text-file-path-or-csv-file-path output-path
hdfs dfs -cat output-path/part-r-00000

Output :
```

```
Program 5 :
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.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class FifthProgram {
      public static class Map extends
Mapper<LongWritable,Text,Text,IntWritable>{
            public void map(LongWritable key,Text value,Context context)
throws IOException,InterruptedException{
                  String[] cols = value.toString().split(",");
                  String gender = cols[2];
                  context.write(new Text(gender), new IntWritable(1));
            }
      }
```

```
      public static class Reduce extends
Reducer<Text,IntWritable,Text,IntWritable>{
            private int totalFemale = 0;
            public void reduce(Text key,Iterable<IntWritable>
values,Context context) throws IOException,InterruptedException{
                  int sum = 0;
                  for(IntWritable value : values) {
                        sum += value.get();
                  }
                  if(key.equals(new Text("Female"))) {
                        totalFemale = sum;
                  }
            }

            public void cleanup(Context context) throws
IOException,InterruptedException{
                  context.write(new Text("Total female voters : "), new
IntWritable(totalFemale));
            }
      }

      public static void main(String args[]) throws Exception {
            Configuration conf = new Configuration();
            Job job = Job.getInstance(conf,"FifthProgram");

            job.setJarByClass(FifthProgram.class);

            job.setMapperClass(Map.class);
            job.setReducerClass(Reduce.class);

            job.setOutputKeyClass(Text.class);
            job.setOutputValueClass(IntWritable.class);

            FileInputFormat.addInputPath(job, new Path(args[0]));
            FileOutputFormat.setOutputPath(job, new Path(args[1]));

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

```
Text File :
1,Divya,Male,20
2,Sumit,Male,20
3,Preksha,Female,20
4,Nikita,Female,20
5,Jishan,Male,20
6,Jhuveriya,Female,20
7,Nisarg,Male,20
8,Meet,Male,20
9,Kirsha,Female,20
10,Karina,Female,20

Commands :
Put file in Hadoop file system :
hdfs dfs -put source destination
```

```
hadoop jar jar-path text-file-path-or-csv-file-path output-path
hdfs dfs -cat output-path/part-r-00000
```

Output :

Program 6 :

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

public class SixthProgram {
      public static class Map extends
Mapper<LongWritable,Text,Text,IntWritable>{
            public void map(LongWritable key,Text value,Context context)
throws IOException,InterruptedException {
                  String cols[] = value.toString().split(",");
                  String reviewID = cols[0];
                  context.write(new Text(reviewID), new IntWritable(1));
            }
      }

      public static class Reduce extends
Reducer<Text,IntWritable,Text,IntWritable>{
            private int total_unique_reviews = 0;
            public void reduce(Text key,Iterable<IntWritable>
values,Context context) throws IOException,InterruptedException{
                  int sum = 0;
                  for(IntWritable value : values) {
                        sum += value.get();
                  }
                  total_unique_reviews++;
                  context.write(key, new IntWritable(sum));
            }

            public void cleanup(Context context) throws
IOException,InterruptedException{
                  context.write(new Text("Total unique reviews : "), new
IntWritable(total_unique_reviews));
            }
```

```
	}

	public static void main(String args[]) throws Exception {
		Configuration conf = new Configuration();
		Job job = Job.getInstance(conf,"SixthProgram");

		job.setJarByClass(SixthProgram.class);
		job.setMapperClass(Map.class);
		job.setReducerClass(Reduce.class);

		job.setOutputKeyClass(Text.class);
		job.setOutputValueClass(IntWritable.class);

		FileInputFormat.addInputPath(job, new Path(args[0]));
		FileOutputFormat.setOutputPath(job, new Path(args[1]));

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

CSV File :
Note : file is large in size as well as rows wise.

Commands :
Put file in Hadoop file system :
hdfs dfs -put source destination
hadoop jar jar-path text-file-path-or-csv-file-path output-path
hdfs dfs -cat output-path/part-r-00000

Output :

Program 7 :

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

public class SeventhProgram {
```

```
      public static class Map extends
Mapper<LongWritable,Text,Text,IntWritable>{
            public void map(LongWritable key,Text value,Context context)
throws IOException,InterruptedException{
                  String col[] = value.toString().split(",");
                  String title = col[1].toString();
                  String genres = col[2].toString();
                  if(genres.contains("Comedy")) {
                        context.write(new Text(title+" : "+genres),new
IntWritable(1));
                  }
                  if(genres.contains("Documentary") &&
title.contains("1995")) {
                        context.write(new Text("Documentry"),new
IntWritable(1));
                  }
                  if(title.contains("Gold")) {
                        context.write(new Text(title),new IntWritable(1));
                  }
                  if(genres.contains("Drama") &&
genres.contains("Romance")) {
                        context.write(new Text(title + " : "+genres),new
IntWritable(1));
                  }
                  if(genres.isEmpty()) {
                        context.write(new Text("Missing"), new
IntWritable(1));
                  }
            }
      }

      public static class Reduce extends
Reducer<Text,IntWritable,Text,IntWritable>{
            private int count = 0;
            private int missing = 0;
            public void reduce(Text key,Iterable<IntWritable>
values,Context context) throws IOException,InterruptedException{
                  int sum = 0;
                  for(IntWritable value : values) {
                        sum += value.get();
                  }
                  context.write(key, new IntWritable(sum));
                  if(key.toString().contains("Documentary")) {
                        count++;
                  }
                  if(key.toString().contains("Documentry")) {
                        missing++;
                  }
            }

            public void cleanup(Context context) throws
IOException,InterruptedException{
                  context.write(new Text("Total documentry movie in 1995 :
"), new IntWritable(count));
                  context.write(new Text("Total missing genres : "), new
IntWritable(missing));
            }
      }
```

```
	public static void main(String args[]) throws Exception {
		Configuration conf = new Configuration();
		Job job = Job.getInstance(conf,"SeventhProgram");

		job.setJarByClass(SeventhProgram.class);
		job.setMapperClass(Map.class);
		job.setReducerClass(Reduce.class);

		job.setOutputKeyClass(Text.class);
		job.setOutputValueClass(IntWritable.class);

		FileInputFormat.addInputPath(job, new Path(args[0]));
		FileOutputFormat.setOutputPath(job, new Path(args[1]));

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

CSV File :
Note : file is large in size as well as rows wise.

Commands :
Put file in Hadoop file system :

```
hdfs dfs -put source destination
hadoop jar jar-path text-file-path-or-csv-file-path output-path
hdfs dfs -cat output-path/part-r-00000
```

Output :