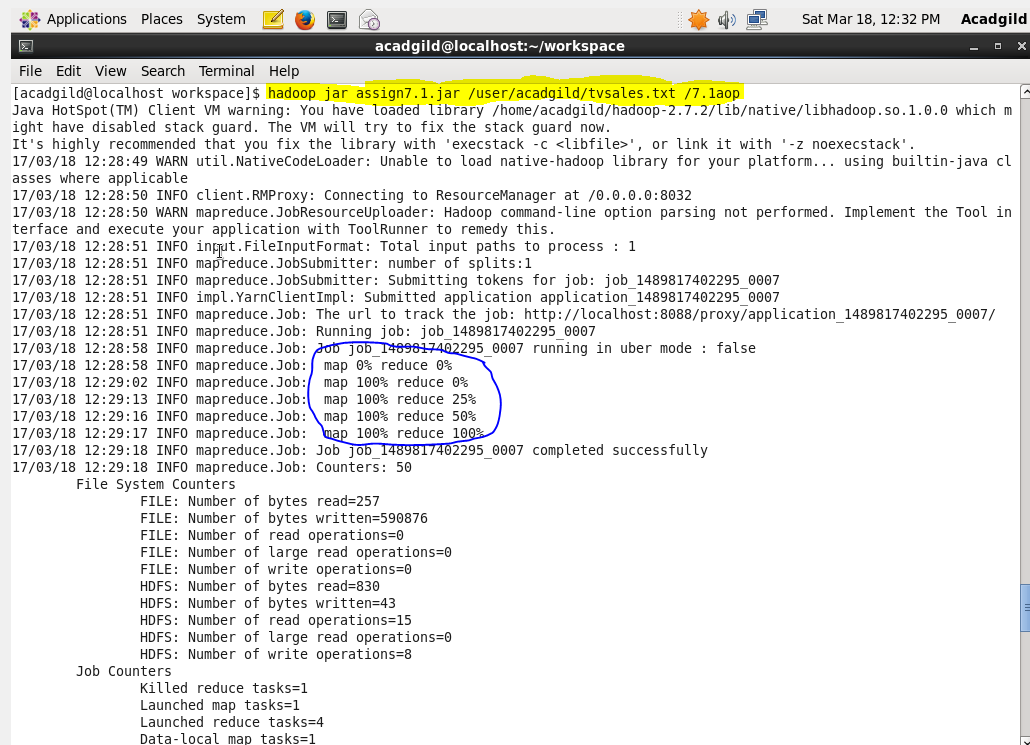
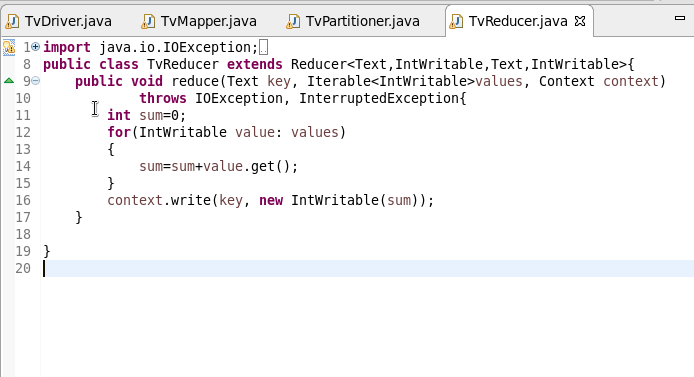
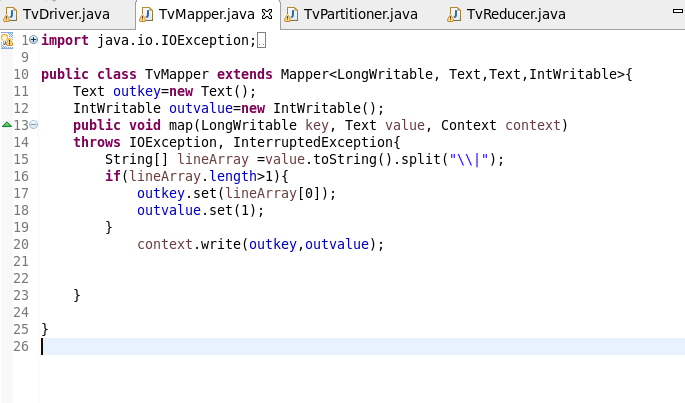
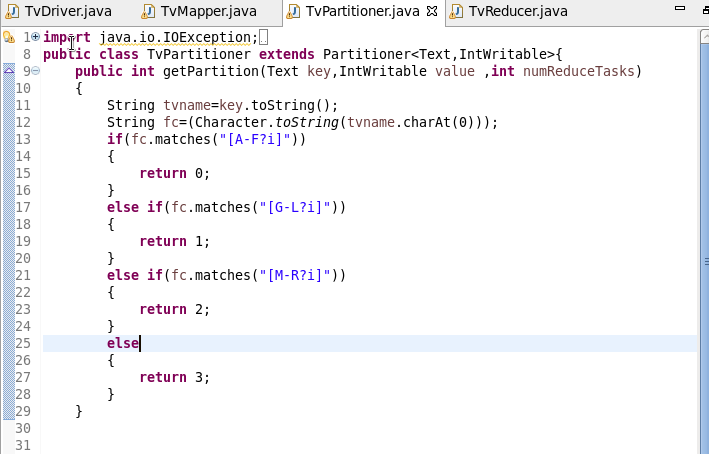
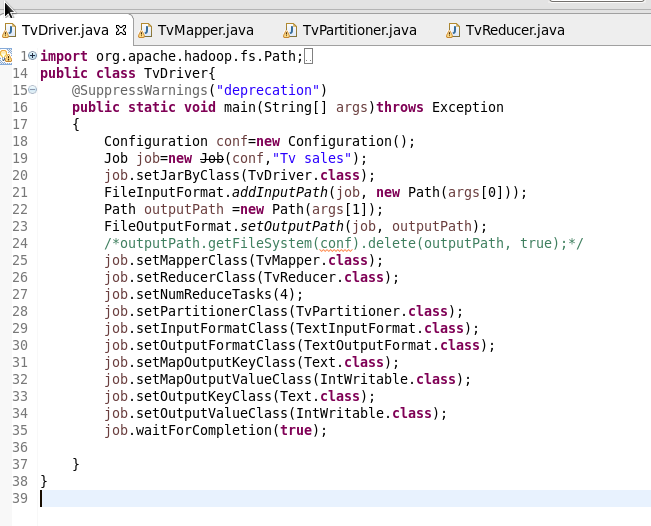
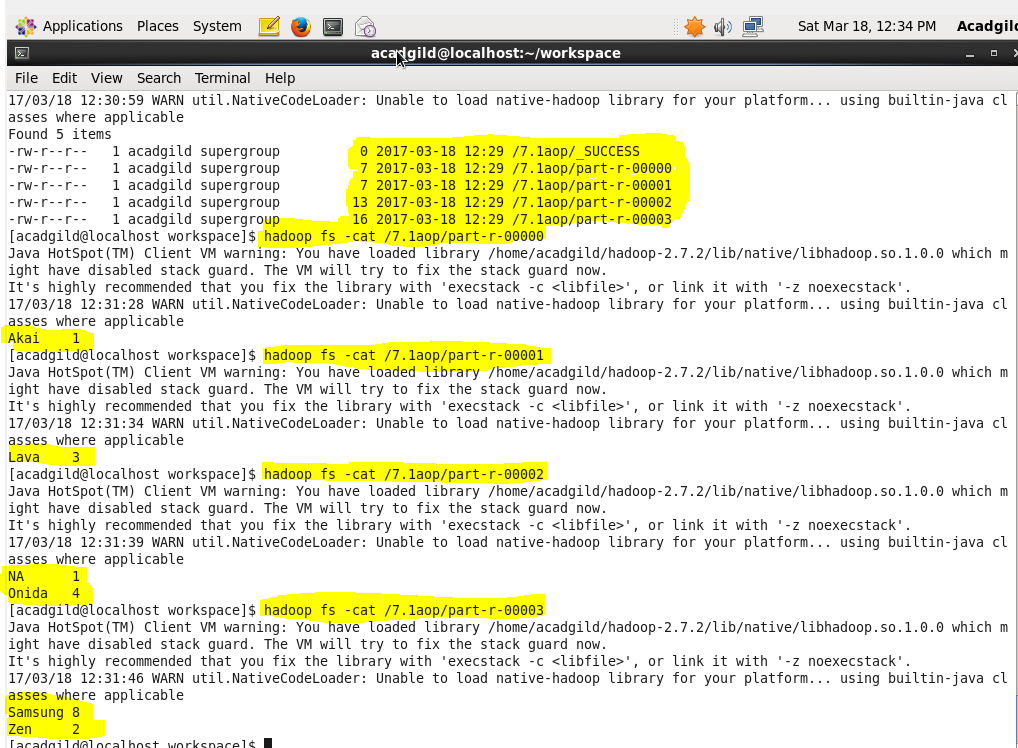
Write a Mapreduce program to modify Sales of different TV Task 2 ( refer session 5 assignment 2 ) to use a custom partitioner with 4 reducers. Make sure that all records whose company name starts with A-F (upper or lower case) should go to 1st reducer, those starting with G-L to 2nd reducer, those starting with M-R to 3rd reducer and others to 4th reducer.

import org.apache.hadoop.fs.Path;  
  
import java.io.IOException;  
  
import org.apache.hadoop.conf.\*;  
import org.apache.hadoop.mapreduce.Job;  
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;  
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;  
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;  
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;  
import org.apache.hadoop.io.IntWritable;  
import org.apache.hadoop.io.NullWritable;  
import org.apache.hadoop.io.Text;  
public class TvDriver{  
    @SuppressWarnings("deprecation")  
    public static void main(String[] args)throws Exception  
    {  
        Configuration conf=new Configuration();  
        Job job=new Job(conf,"Tv sales");  
        job.setJarByClass(TvDriver.class);  
        FileInputFormat.addInputPath(job, new Path(args[0]));  
        Path outputPath =new Path(args[1]);  
        FileOutputFormat.setOutputPath(job, outputPath);  
        /\*outputPath.getFileSystem(conf).delete(outputPath, true);\*/  
        job.setMapperClass(TvMapper.class);  
        job.setReducerClass(TvReducer.class);  
        job.setNumReduceTasks(4);  
        job.setPartitionerClass(TvPartitioner.class);  
        job.setInputFormatClass(TextInputFormat.class);  
        job.setOutputFormatClass(TextOutputFormat.class);  
        job.setMapOutputKeyClass(Text.class);  
        job.setMapOutputValueClass(IntWritable.class);  
        job.setOutputKeyClass(Text.class);  
        job.setOutputValueClass(IntWritable.class);  
        job.waitForCompletion(true);  
  
    }  
}  
import java.io.IOException;  
import java.util.StringTokenizer;  
  
import org.apache.hadoop.io.IntWritable;  
import org.apache.hadoop.io.LongWritable;  
import org.apache.hadoop.io.NullWritable;  
import org.apache.hadoop.io.Text;  
import org.apache.hadoop.mapreduce.\*;  
  
public class TvMapper extends Mapper<LongWritable, Text,Text,IntWritable>{  
    Text outkey=new Text();  
    IntWritable outvalue=new IntWritable();  
    public void map(LongWritable key, Text value, Context context)  
    throws IOException, InterruptedException{  
        String[] lineArray =value.toString().split("\\|");  
        if(lineArray.length>1){  
            outkey.set(lineArray[0]);  
            outvalue.set(1);  
        }  
            context.write(outkey,outvalue);  
  
  
    }  
  
}  
import java.io.IOException;  
import org.apache.hadoop.io.IntWritable;  
import org.apache.hadoop.io.LongWritable;  
import org.apache.hadoop.io.NullWritable;  
import org.apache.hadoop.io.Text;  
import org.apache.hadoop.mapreduce.\*;  
import org.apache.hadoop.mapreduce.Mapper.Context;  
public class TvPartitioner extends Partitioner<Text,IntWritable>{  
    public int getPartition(Text key,IntWritable value ,int numReduceTasks)  
    {  
        String tvname=key.toString();  
        String fc=(Character.toString(tvname.charAt(0)));  
        if(fc.matches("[A-F?i]"))  
        {  
            return 0;  
        }  
        else if(fc.matches("[G-L?i]"))  
        {  
            return 1;  
        }  
        else if(fc.matches("[M-R?i]"))  
        {  
            return 2;  
        }  
        else  
        {  
            return 3;  
        }  
    }  
  
  
  
  
  
  
  
}  
import java.io.IOException;  
import org.apache.hadoop.io.IntWritable;  
import org.apache.hadoop.io.LongWritable;  
import org.apache.hadoop.io.NullWritable;  
import org.apache.hadoop.io.Text;  
import org.apache.hadoop.mapreduce.\*;  
import org.apache.hadoop.mapreduce.Mapper.Context;  
public class TvReducer extends Reducer<Text,IntWritable,Text,IntWritable>{  
    public void reduce(Text key, Iterable<IntWritable>values, Context context)  
            throws IOException, InterruptedException{  
        int sum=0;  
        for(IntWritable value: values)  
        {  
            sum=sum+value.get();  
        }  
        context.write(key, new IntWritable(sum));  
    }}  
  


Modify Sales of different TV Task 2 (refer session 5, assignment 2) to take advantage of Combiner.

import org.apache.hadoop.fs.Path;  
import org.apache.hadoop.conf.\*;  
import org.apache.hadoop.mapreduce.Job;  
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;  
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;  
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;  
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;  
import org.apache.hadoop.io.IntWritable;  
import org.apache.hadoop.io.NullWritable;  
import org.apache.hadoop.io.Text;  
public class TvDriver{  
    @SuppressWarnings("deprecation")  
    public static void main(String[] args)throws Exception  
    {  
        Configuration conf=new Configuration();  
        Job job=new Job(conf,"Tv sales");  
        job.setJarByClass(TvDriver.class);  
        FileInputFormat.addInputPath(job, new Path(args[0]));  
        Path outputPath =new Path(args[1]);  
        FileOutputFormat.setOutputPath(job, outputPath);  
        outputPath.getFileSystem(conf).delete(outputPath, true);  
        job.setMapperClass(TvMapper.class);  
        job.setReducerClass(TvReducer.class);  
        job.setCombinerClass(TvReducer.class);  
        job.setInputFormatClass(TextInputFormat.class);  
        job.setOutputFormatClass(TextOutputFormat.class);  
        job.setMapOutputKeyClass(Text.class);  
        job.setMapOutputValueClass(IntWritable.class);  
        job.setOutputKeyClass(Text.class);  
        job.setOutputValueClass(IntWritable.class);  
        job.waitForCompletion(true);  
  
    }  
}  
    import java.io.IOException;  
import java.util.StringTokenizer;  
  
import org.apache.hadoop.io.IntWritable;  
import org.apache.hadoop.io.LongWritable;  
import org.apache.hadoop.io.NullWritable;  
import org.apache.hadoop.io.Text;  
import org.apache.hadoop.mapreduce.\*;  
  
public class TvMapper extends Mapper<LongWritable, Text,Text,IntWritable>{  
    Text outkey=new Text();  
    IntWritable outvalue=new IntWritable();  
    public void map(LongWritable key, Text value, Context context)  
    throws IOException, InterruptedException{  
        String[] lineArray =value.toString().split("\\|");  
        if(lineArray.length>1){  
            outkey.set(lineArray[0]);  
            outvalue.set(1);  
        }  
            context.write(outkey,outvalue);  
  
  
    }  
  
}import java.io.IOException;  
import org.apache.hadoop.io.IntWritable;  
import org.apache.hadoop.io.LongWritable;  
import org.apache.hadoop.io.NullWritable;  
import org.apache.hadoop.io.Text;  
import org.apache.hadoop.mapreduce.\*;  
import org.apache.hadoop.mapreduce.Mapper.Context;  
public class TvReducer extends Reducer<Text,IntWritable,Text,IntWritable>{  
    public void reduce(Text key, Iterable<IntWritable>values, Context context)  
            throws IOException, InterruptedException{  
        int sum=0;  
        for(IntWritable value: values)  
        {  
            sum=sum+value.get();  
        }  
        context.write(key, new IntWritable(sum));  
    }  
  
}

