Assignment -3.1

Task1: Write a Map Reduce program to filter out the invalid records. Map only job will fit for this context.

Answer1:

Java code screen shots are as below.(Mapper code + Driver code)

InvalidMapper.java | InvalidDriver.java

```java
package SalesDetails;

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;
import org.apache.hadoop.mapreduce.Mapper.Context;

import java.util.*;

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

	private Text record = new Text();

	public void map(LongWritable key, Text value, Context context)
	throws IOException, InterruptedException {
	    int j = 0;
	    String line = value.toString();
	    record = value;
	    String[] lineArray = record.toString().split("\\|");

	    for (int i=0; i< lineArray.length; i++) {
	        if (lineArray[i].equals("NA")) {
	            j=1;
	        }
	    }
	    if (j!=1) {
	        context.write(record, null);
	    }
	}
}
```

InvalidMapper.java | InvalidDriver.java

```java
package SalesDetails;

import java.io.IOException;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
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.mapreduce.lib.output.TextOutputFormat;

public class InvalidDriver {

    public static void main(String[] args) throws Exception {
        if (args.length != 2) {
            System.err.println("Usage: InvalidDriver <input path> <output path>");
            System.exit(-1);
         }

            //Job Related Configurations
            Configuration conf = new Configuration();
            Job job = new Job(conf, "Invalid TV records");
            job.setJarByClass(InvalidDriver.class);


            // Specify the number of reducer to 2
            job.setNumReduceTasks(0);

            //Provide paths to pick the input file for the job
            FileInputFormat.setInputPaths(job, new Path(args[0]));


            //Provide paths to pick the output file for the job, and delete it if already present
            Path outputPath = new Path(args[1]);
            FileOutputFormat.setOutputPath(job, outputPath);
            outputPath.getFileSystem(conf).delete(outputPath, true);

            //To set the mapper and reducer of this job
            job.setMapperClass(InvalidMapper.class);
            // job.setReducerClass(WordCountReducer.class);

            //Set the combiner
            //  job.setCombinerClass(WordCountReducer.class);

            //set the input and output format class
            job.setInputFormatClass(TextInputFormat.class);
            job.setOutputFormatClass(TextOutputFormat.class);

            //set up the output key and value classes
            job.setOutputKeyClass(Text.class);
            job.setOutputValueClass(IntWritable.class);

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

Input Command:

hadoop jar InvalidTVrec.jar /television.txt /TVInvalidout

```
[acadgild@localhost ~]$ hadoop jar InvalidTVrec.jar /television.txt /TVInvalidout
18/03/12 22:37:48 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes whe
re applicable
18/03/12 22:37:50 INFO client.RMProxy: Connecting to ResourceManager at localhost/127.0.0.1:8032
18/03/12 22:37:51 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface a
nd execute your application with ToolRunner to remedy this.
18/03/12 22:37:51 INFO input.FileInputFormat: Total input paths to process : 1
18/03/12 22:37:51 INFO mapreduce.JobSubmitter: number of splits:1
18/03/12 22:37:51 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1520874036548_0002
18/03/12 22:37:52 INFO impl.YarnClientImpl: Submitted application application_1520874036548_0002
18/03/12 22:37:52 INFO mapreduce.Job: The url to track the job: http://localhost:8088/proxy/application_1520874036548_0002/
18/03/12 22:37:52 INFO mapreduce.Job: Running job: job_1520874036548_0002
18/03/12 22:38:01 INFO mapreduce.Job: Job job_1520874036548_0002 running in uber mode : false
18/03/12 22:38:01 INFO mapreduce.Job:  map 0% reduce 0%
18/03/12 22:38:07 INFO mapreduce.Job:  map 100% reduce 0%
18/03/12 22:38:08 INFO mapreduce.Job: Job job_1520874036548_0002 completed successfully
18/03/12 22:38:08 INFO mapreduce.Job: Counters: 30
        File System Counters
                FILE: Number of bytes read=0
                FILE: Number of bytes written=107371
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=834
                HDFS: Number of bytes written=646
                HDFS: Number of read operations=5
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=4565
```

Output: Since this is a mapper only job – the mapper output is checked.

Also here all the records with “NA” has been removed as a part of mapper logic.

```
[acadgild@localhost ~]$ hadoop fs -cat /TVInvalidout/part-m-00000
18/03/12 22:38:29 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes whe
re applicable
Samsung|Optima|14|Madhya Pradesh|132401|14200
Onida|Lucid|18|Uttar Pradesh|232401|16200
Akai|Decent|16|Kerala|922401|12200
Lava|Attention|20|Assam|454601|24200
Zen|Super|14|Maharashtra|619082|9200
Samsung|Optima|14|Madhya Pradesh|132401|14200
Onida|Lucid|18|Uttar Pradesh|232401|16200
Onida|Decent|14|Uttar Pradesh|232401|16200
Lava|Attention|20|Assam|454601|24200
Zen|Super|14|Maharashtra|619082|9200
Samsung|Optima|14|Madhya Pradesh|132401|14200
Samsung|Decent|16|Kerala|922401|12200
Lava|Attention|20|Assam|454601|24200
Samsung|Super|14|Maharashtra|619082|9200
Samsung|Super|14|Maharashtra|619082|9200
Samsung|Super|14|Maharashtra|619082|9200
```

Task2: Write a Map Reduce program to calculate the total units sold for each company.

Answer2:

Java code screen shots are as below.(Mapper code + Reducer code + Driver code)

CompanyCountMapper.java | CompanyCountReducer.java | CompanyCountDriver.java

```java
package CompanyCount;

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;
import org.apache.hadoop.mapreduce.Mapper.Context;

import java.util.*;

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

	private final static IntWritable one = new IntWritable(1);
	private Text record = new Text();

	public void map(LongWritable key, Text value, Context context)
	throws IOException, InterruptedException {
		int j=0;
		String line = value.toString();
		record = value;
		String[] lineArray = record.toString().split("\\|");

		if (lineArray[0].equals(null) || lineArray[0].equals("NA")) {
			j=1;
		}

		if (j!=1) {
			Text t1 = new Text(lineArray[0]);
			context.write(t1, one);
		}
	}
}
```

CompanyCountMapper.java | CompanyCountReducer.java | Compa

```java
package CompanyCount;

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;


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

  @Override
  public void reduce(Text key, Iterable<IntWritable> values,
      Context context)
      throws IOException, InterruptedException {
      System.out.println("From The Reducer=>"+key) ;

      int sum = 0;
      for (IntWritable value : values) {
        sum+=value.get();
       }
       context.write(key, new IntWritable(sum));
  }
}
```

CompanyCountMapper.java | CompanyCountReducer.java | CompanyCountDriver.java

```java
package CompanyCount;

import java.io.IOException;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
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.mapreduce.lib.output.TextOutputFormat;

import CompanyCount.CompanyCountDriver;

public class CompanyCountDriver {

    public static void main(String[] args) throws Exception {
        // TODO Auto-generated method stub
        if (args.length != 2) {
            System.err.println("Usage: CompanyCountDriver <input path> <output path>");
            System.exit(-1);
          }

        //Job Related Configurations
        Configuration conf = new Configuration();
        Job job = new Job(conf, "Total Units sold for each Company");
        job.setJarByClass(CompanyCountDriver.class);

        // Specify the number of reducer to 1
        job.setNumReduceTasks(1);

        //Provide paths to pick the input file for the job
        FileInputFormat.setInputPaths(job, new Path(args[0]));

        //Provide paths to pick the output file for the job, and delete it if already present
        Path outputPath = new Path(args[1]);
        FileOutputFormat.setOutputPath(job, outputPath);
        outputPath.getFileSystem(conf).delete(outputPath, true);

        //To set the mapper and reducer of this job
        job.setMapperClass(CompanyCount.CompanyCountMapper.class);
        job.setReducerClass(CompanyCount.CompanyCountReducer.class);

        //set the input and output format class
        job.setInputFormatClass(TextInputFormat.class);
        job.setOutputFormatClass(TextOutputFormat.class);

        //set up the output key and value classes
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);

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

}
```

Input Command:

hadoop jar CompanyCount.jar /television.txt /CompCountout

```
[acadgild@localhost ~]$ hadoop jar CompanyCount.jar /television.txt /CompCountout
18/03/12 22:55:48 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes whe
re applicable
18/03/12 22:55:49 INFO client.RMProxy: Connecting to ResourceManager at localhost/127.0.0.1:8032
18/03/12 22:55:50 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface a
nd execute your application with ToolRunner to remedy this.
18/03/12 22:55:50 INFO input.FileInputFormat: Total input paths to process : 1
18/03/12 22:55:50 INFO mapreduce.JobSubmitter: number of splits:1
18/03/12 22:55:51 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1520874036548_0004
18/03/12 22:55:51 INFO impl.YarnClientImpl: Submitted application application_1520874036548_0004
18/03/12 22:55:51 INFO mapreduce.Job: The url to track the job: http://localhost:8088/proxy/application_1520874036548_0004/
18/03/12 22:55:51 INFO mapreduce.Job: Running job: job_1520874036548_0004
18/03/12 22:56:00 INFO mapreduce.Job: Job job_1520874036548_0004 running in uber mode : false
18/03/12 22:56:00 INFO mapreduce.Job:  map 0% reduce 0%
18/03/12 22:56:07 INFO mapreduce.Job:  map 100% reduce 0%
18/03/12 22:56:15 INFO mapreduce.Job:  map 100% reduce 100%
18/03/12 22:56:15 INFO mapreduce.Job: Job job_1520874036548_0004 completed successfully
18/03/12 22:56:15 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=216
                FILE: Number of bytes written=215815
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=834
                HDFS: Number of bytes written=38
                HDFS: Number of read operations=6
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=4371
                Total time spent by all reduces in occupied slots (ms)=4883
                Total time spent by all map tasks (ms)=4371
```

Output: Here the Company counts are mentioned for valid records only.

```
[acadgild@localhost ~]$ hadoop fs -cat /CompCountout/part-r-00000
18/03/12 22:56:38 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes whe
re applicable
Akai    1
Lava    3
Onida   4
Samsung 7
Zen     2
```

Task3: Write a Map Reduce program to calculate the total units sold in each state for Onida company.

Answer3:

Java code screen shots are as below.(Mapper code + Reducer code + Driver code)

StateCountMapper.java | StateCountReducer.java | StateCountDriver.java

```java
package StateCount;

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;
import org.apache.hadoop.mapreduce.Mapper.Context;

import java.util.*;

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

      private final static IntWritable one = new IntWritable(1);
      private Text record = new Text();

      public void map(LongWritable key, Text value, Context context)
      throws IOException, InterruptedException {
          int j=0;
          String line = value.toString();
          record = value;
          String[] lineArray = record.toString().split("\\|");

          if (lineArray[0].equals("Onida")) {
              // Gets the State name only for Onida company.
              Text t1 = new Text(lineArray[3]);
              context.write(t1, one);
          }
      }
}
```

StateCountMapper.java | StateCountReducer.java | StateCountDriver.j

```java
package StateCount;

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;


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

  @Override
  public void reduce(Text key, Iterable<IntWritable> values,
      Context context)
      throws IOException, InterruptedException {
      System.out.println("From The Reducer=>"+key) ;

      int sum = 0;
      for (IntWritable value : values) {
        sum+=value.get();
       }
       context.write(key, new IntWritable(sum));
  }
}
```

StateCountMapper.java | StateCountReducer.java | StateCountDriver.java

```java
package StateCount;

import java.io.IOException;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
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.mapreduce.lib.output.TextOutputFormat;

import StateCount.StateCountDriver;

public class StateCountDriver {

    public static void main(String[] args) throws IOException, ClassNotFoundException, InterruptedException {
        // TODO Auto-generated method stub
        if (args.length != 2) {
            System.err.println("Usage: StateCountDriver <input path> <output path>");
            System.exit(-1);
          }

        //Job Related Configurations
        Configuration conf = new Configuration();
        Job job = new Job(conf, "Total Units sold in each State for Onida Company");
        job.setJarByClass(StateCountDriver.class);

        // Specify the number of reducer to 2
        job.setNumReduceTasks(1);

        //Provide paths to pick the input file for the job
        FileInputFormat.setInputPaths(job, new Path(args[0]));

          //Provide paths to pick the output file for the job, and delete it if already present
          Path outputPath = new Path(args[1]);
          FileOutputFormat.setOutputPath(job, outputPath);
          outputPath.getFileSystem(conf).delete(outputPath, true);

          //To set the mapper and reducer of this job
          job.setMapperClass(StateCount.StateCountMapper.class);
          job.setReducerClass(StateCount.StateCountReducer.class);

          //set the input and output format class
          job.setInputFormatClass(TextInputFormat.class);
          job.setOutputFormatClass(TextOutputFormat.class);

          //set up the output key and value classes
          job.setOutputKeyClass(Text.class);
          job.setOutputValueClass(IntWritable.class);

          //execute the job
          System.exit(job.waitForCompletion(true) ? 0 : 1);

      }

}
```

Input Command:

hadoop jar StateCount.jar /television.txt /StateCountout

```
[acadgild@localhost ~]$ hadoop jar StateCount.jar /television.txt /StateCountout
18/03/12 23:07:29 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes whe
re applicable
18/03/12 23:07:30 INFO client.RMProxy: Connecting to ResourceManager at localhost/127.0.0.1:8032
18/03/12 23:07:31 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface a
nd execute your application with ToolRunner to remedy this.
18/03/12 23:07:31 INFO input.FileInputFormat: Total input paths to process : 1
18/03/12 23:07:31 INFO mapreduce.JobSubmitter: number of splits:1
18/03/12 23:07:32 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1520874036548_0005
18/03/12 23:07:32 INFO impl.YarnClientImpl: Submitted application application_1520874036548_0005
18/03/12 23:07:32 INFO mapreduce.Job: The url to track the job: http://localhost:8088/proxy/application_1520874036548_0005/
18/03/12 23:07:32 INFO mapreduce.Job: Running job: job_1520874036548_0005
18/03/12 23:07:41 INFO mapreduce.Job: Job job_1520874036548_0005 running in uber mode : false
18/03/12 23:07:41 INFO mapreduce.Job:  map 0% reduce 0%
18/03/12 23:07:49 INFO mapreduce.Job:  map 100% reduce 0%
18/03/12 23:07:58 INFO mapreduce.Job:  map 100% reduce 100%
18/03/12 23:07:58 INFO mapreduce.Job: Job job_1520874036548_0005 completed successfully
18/03/12 23:07:58 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=79
                FILE: Number of bytes written=215557
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=834
                HDFS: Number of bytes written=25
                HDFS: Number of read operations=6
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=4680
                Total time spent by all reduces in occupied slots (ms)=5589
                Total time spent by all map tasks (ms)=4680
```

Output: Here the counts are only for valid records having company name as “Onida”.

```
[acadgild@localhost ~]$ hadoop fs -cat /StateCountout/part-r-00000
18/03/12 23:08:11 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes whe
re applicable
Kerala  1
Uttar Pradesh   3
[acadgild@localhost ~]$
```