APPENDIX

Code for Twitter Follower Count

```
import java.io.IOException;
import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.util.*;

public class TwitterFollowerCounter {

  public static class Map extends MapReduceBase implements Mapper<LongWritable, Text, Text, IntWritable> {
    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();


    public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output, Reporter reporter) throws IOException {
      String line = value.toString();
      StringTokenizer tokenizer = new StringTokenizer(line);
      if (tokenizer.hasMoreTokens()) {
        word.set(tokenizer.nextToken());
        output.collect(word, one);
      }
    }
  }


  public static class Reduce extends MapReduceBase implements Reducer<Text, IntWritable, Text, IntWritable> {
    public void reduce(Text key, Iterator<IntWritable> values, OutputCollector<Text, IntWritable> output, Reporter reporter) throws IOException
{
      int sum = 0;
      while (values.hasNext()) {
        sum += values.next().get();
      }
      output.collect(key, new IntWritable(sum));
    }
  }

public static void main(String[] args) throws Exception {
    JobConf conf = new JobConf(TwitterFollowerCounter.class);
    conf.setJobName("TwitterFollowerCounter");
conf.setOutputKeyClass(Text.class);
    conf.setOutputValueClass(IntWritable.class);
    conf.setMapperClass(Map.class);
    //conf.setCombinerClass(Reduce.class);
    conf.setReducerClass(Reduce.class);
    conf.setInputFormat(TextInputFormat.class);
    conf.setOutputFormat(TextOutputFormat.class);
    FileInputFormat.setInputPaths(conf, new Path(args[0]));
    FileOutputFormat.setOutputPath(conf, new Path(args[1]));
JobClient.runJob(conf);
  }
}
```

Code for Twitter Follower Count Group by Range

```
import java.io.IOException;
import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
```

```
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.util.*;

public class TwitterFollowerCounterGroupByRanges {


public static class Map extends MapReduceBase implements Mapper<LongWritable, Text, IntWritable, IntWritable> {
private IntWritable key = new IntWritable();
private final static IntWritable one = new IntWritable(1);
private int makeRangeKeyFromCountFollowers(int count_followers) { // 22 -> 100
int x = 1;
while (x <= count_followers) {
x *= 10;
}
return x;
}
public void map(LongWritable _key, Text value, OutputCollector<IntWritable, IntWritable> output, Reporter reporter) throws IOException {
String line = value.toString();
String[] twitter_id_and_count_followers = line.split("\t");
if (twitter_id_and_count_followers.length == 2) {
Integer _count_followers = Integer.parseInt(twitter_id_and_count_followers[1]);
key.set(makeRangeKeyFromCountFollowers(_count_followers));
output.collect(key, one);
}
}
}
public static class Reduce extends MapReduceBase implements Reducer<IntWritable, IntWritable, IntWritable, IntWritable> {
private IntWritable val = new IntWritable();
public void reduce(IntWritable key, Iterator<IntWritable> values, OutputCollector<IntWritable, IntWritable> output, Reporter reporter) throws
output.collect(key, new IntWritable(sum));
}
}
public static void main(String[] args) throws Exception {
JobConf conf = new JobConf(TwitterFollowerCounterGroupByRanges.class);
conf.setJobName("TwitterFollowerCounterGroupByRanges");
conf.setOutputKeyClass(IntWritable.class);
conf.setOutputValueClass(IntWritable.class);
conf.setMapperClass(Map.class);
//conf.setCombinerClass(Reduce.class);
conf.setReducerClass(Reduce.class);
conf.setInputFormat(TextInputFormat.class);
conf.setOutputFormat(TextOutputFormat.class);
FileInputFormat.setInputPaths(conf, new Path(args[0]));
FileOutputFormat.setOutputPath(conf, new Path(args[1]));
JobClient.runJob(conf);
}
}
```

Code for Twitter Sentiment Analysis

```
import java.io.*;
import java.io.InputStreamReader;
import java.net.URI;
import java.util.HashMap;
import org.apache.hadoop..*;
import org.apache.hadoop.util.Tool;
import org.json.simple.JSONObject;
import org.json.simple.parser.JSONParser;

public class Sentiment_Analysis extends Configured implements Tool {
public static class Map extends Mapper<LongWritable, Text, Text, Text>{
private URI[] files;
private HashMap<String,String> AFINN_map = new HashMap<String,String>();

public void setup(Context context) throws IOException
{
files = DistributedCache.getCacheFiles(context.getConfiguration());
System.out.println("files:"+ files);
Path path = new Path(files[0]);
```

```
FileSystem fs = FileSystem.get(context.getConfiguration());
FSDataInputStream in = fs.open(path);
BufferedReader br = new BufferedReader(new InputStreamReader(in));
String line="";
while((line = br.readLine())!=null)
{
String splits[] = line.split("\t");
AFINN_map.put(splits[0], splits[1]);
}
br.close();
in.close();
}
public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException{
String name;
String twt;
String line = value.toString();
String[] tuple = line.split("\\n");
JSONParser jsonParser = new JSONParser();

try{
for(int i=0;i<tuple.length; i++){
JSONObject obj =(JSONObject) jsonParser.parse(line);
String tweet_id = (String) obj.get("id_str");
String tweet_text=(String) obj.get("text");
twt=(String) obj.get("text");
String[] splits = twt.toString().split(" ");
int sentiment_sum=0;
for(String word:splits){
if(AFINN_map.containsKey(word))
{
Integer x=new Integer(AFINN_map.get(word));
sentiment_sum+=x;
}
}
context.write(new Text(tweet_id),new Text(tweet_text+"\t----->\t"+new Text(Integer.toString(sentiment_sum))));
}
}catch(Exception e){
e.printStackTrace();
}
public static class Reduce extends Reducer<Text,Text,Text,Text>{
public void reduce(Text key, Text value, Context context) throws IOException, InterruptedException{
}
public static void main(String[] args) throws Exception
{
ToolRunner.run(new Sentiment_Analysis(),args);
}
public int run(String[] args) throws Exception {
Configuration conf = new Configuration();
if (args.length != 2) {
System.err.println("Usage: Parse <in> <out>");
System.exit(2);
}
DistributedCache.addCacheFile(new URI("/AFINN.txt"),conf);
Job job = new Job(conf, "SentimentAnalysis");
job.setJarByClass(Sentiment_Analysis .class);
job.setMapperClass(Map.class);
job.setReducerClass(Reduce.class);
job.setMapOutputKeyClass(Text.class);
job.setMapOutputValueClass(Text.class);
job.setOutputKeyClass(NullWritable.class);
job.setOutputValueClass(Text.class);
job.setInputFormatClass(TextInputFormat.class);
job.setOutputFormatClass(TextOutputFormat.class);
FileInputFormat.addInputPath(job, new Path(args[0]));
FileOutputFormat.setOutputPath(job, new Path(args[1]));
System.exit(job.waitForCompletion(true) ? 0 : 1);
return 0;
}
}
```