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

private Text status = new Text();

private final static IntWritable addOne = new IntWritable(1);

/\*\*

\* Returns the SMS status code and its count

\*/

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

throws java.io.IOException, InterruptedException {

//655209;1;796764372490213;804422938115889;6 is the Sample record format

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

// If record is of SMS CDR

if (Integer.parseInt(line[1]) == 1) {

status.set(line[4]);

context.write(status, addOne);

}

}

}

================

public class SMSCDRReducer extends

Reducer<Text, IntWritable, Text, IntWritable> {

protected void reduce(Text key, Iterable<IntWritable> values, Context context) throws java.io.IOException, InterruptedException {

int sum = 0;

for (IntWritable value : values) {

sum += value.get();

}

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

}

}

========================

import java.util.ArrayList;

import java.util.List;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mrunit.mapreduce.MapDriver;

import org.apache.hadoop.mrunit.mapreduce.MapReduceDriver;

import org.apache.hadoop.mrunit.mapreduce.ReduceDriver;

import org.junit.Before;

import org.junit.Test;

public class SMSCDRMapperReducerTest {

MapDriver<LongWritable, Text, Text, IntWritable> mapDriver;

ReduceDriver<Text, IntWritable, Text, IntWritable> reduceDriver;

MapReduceDriver<LongWritable, Text, Text, IntWritable, Text, IntWritable> mapReduceDriver;

@Before

public void setUp() {

SMSCDRMapper mapper = new SMSCDRMapper();

SMSCDRReducer reducer = new SMSCDRReducer();

mapDriver = MapDriver.newMapDriver(mapper);

reduceDriver = ReduceDriver.newReduceDriver(reducer);

mapReduceDriver = MapReduceDriver.newMapReduceDriver(mapper, reducer);

}

@Test

public void testMapper() {

mapDriver.withInput(new LongWritable(), new Text(

"655209;1;796764372490213;804422938115889;6"));

mapDriver.withOutput(new Text("6"), new IntWritable(1));

mapDriver.runTest();

}

@Test

public void testReducer() {

List<IntWritable> values = new ArrayList<IntWritable>();

values.add(new IntWritable(1));

values.add(new IntWritable(1));

reduceDriver.withInput(new Text("6"), values);

reduceDriver.withOutput(new Text("6"), new IntWritable(2));

reduceDriver.runTest();

}

@Test

public void testMapReduce() {

mapReduceDriver.withInput(new LongWritable(), new Text(

"655209;1;796764372490213;804422938115889;6"));

List<IntWritable> values = new ArrayList<IntWritable>();

values.add(new IntWritable(1));

values.add(new IntWritable(1));

mapReduceDriver.withOutput(new Text("6"), new IntWritable(2));

mapReduceDriver.runTest();

}

}

==============

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

private Text status = new Text();

private final static IntWritable addOne = new IntWritable(1);

static enum CDRCounter {

NonSMSCDR;

};

/\*\*

\* Returns the SMS status code and its count

\*/

protected void map(LongWritable key, Text value, Context context) throws java.io.IOException, InterruptedException {

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

// If record is of SMS CDR

if (Integer.parseInt(line[1]) == 1) {

status.set(line[4]);

context.write(status, addOne);

} else {// CDR record is not of type SMS so increment the counter

context.getCounter(CDRCounter.NonSMSCDR).increment(1);

}

}

}