**Problem:**

We have a dataset of sales of different TV sets across different locations.

Records look like:

Samsung|Optima|14|Madhya Pradesh|132401|14200

The fields are arranged like:

Company Name|Product Name|Size in inches|State|Pin Code|Price

There are some invalid records which contain 'NA' in either Company Name or Product Name.

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

context.

**Solution**

Copying the data file in Hadoop file system

[cloudera@quickstart ~]$ hadoop fs -put Desktop/television.txt /Ajay

**Data Files**

[cloudera@quickstart ~]$ hadoop fs -cat /Ajay/television.txt

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

Onida|NA|16|Kerala|922401|12200

Lava|Attention|20|Assam|454601|24200

Zen|Super|14|Maharashtra|619082|9200

Samsung|Optima|14|Madhya Pradesh|132401|14200

NA|Lucid|18|Uttar Pradesh|232401|16200

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[cloudera@quickstart ~]$

**Mapper Program**

**package** com.test.task2;

**import** java.io.IOException;

**import** org.apache.hadoop.io.IntWritable;

//import org.apache.hadoop.io.IntWritable;

**import** org.apache.hadoop.io.LongWritable;

**import** org.apache.hadoop.io.Text;

**import** org.apache.hadoop.mapreduce.\*;

**public** **class** Task1Mapper **extends** Mapper<LongWritable, Text, LongWritable, Text> {

**public** **void** map(LongWritable key, Text value, Context context)

**throws** IOException, InterruptedException {

//String[] lineArray = value.toString().split("|");

//Text errorPattern=" ";

String line = value.toString();

String[] lineArray = line.split("\\W+");

String compName= lineArray[0];

String prodName= lineArray[1];

//while (lineArray.)

{

**if**(compName.equals("NA") )

{

;// do nothing

}

**else** **if** (prodName.equals("NA"))

{

;// do nothing

}

**else**

{

context.write(key, value);

}

}

}

}

**Driver Program**

**package** com.test.task2;

**import** org.apache.hadoop.fs.Path;

**import** org.apache.hadoop.conf.\*;

**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** org.apache.hadoop.io.IntWritable;

**import** org.apache.hadoop.io.LongWritable;

**import** org.apache.hadoop.io.Text;

**public** **class** Task1 {

@SuppressWarnings("deprecation")

**public** **static** **void** main(String[] args) **throws** Exception {

Configuration conf = **new** Configuration();

Job job = **new** ~~Job~~(conf, "DemoTask1");

job.setJarByClass(Task1.**class**);

job.setMapOutputKeyClass(LongWritable.**class**);

job.setMapOutputValueClass(Text.**class**);

job.setOutputKeyClass(LongWritable.**class**);

job.setOutputValueClass(Text.**class**);

job.setMapperClass(Task1Mapper.**class**);

//job.setReducerClass(Task1Reducer.class);

job.setInputFormatClass(TextInputFormat.**class**);

job.setOutputFormatClass(TextOutputFormat.**class**);

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

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

/\*

Path out=new Path(args[1]);

out.getFileSystem(conf).delete(out);

\*/

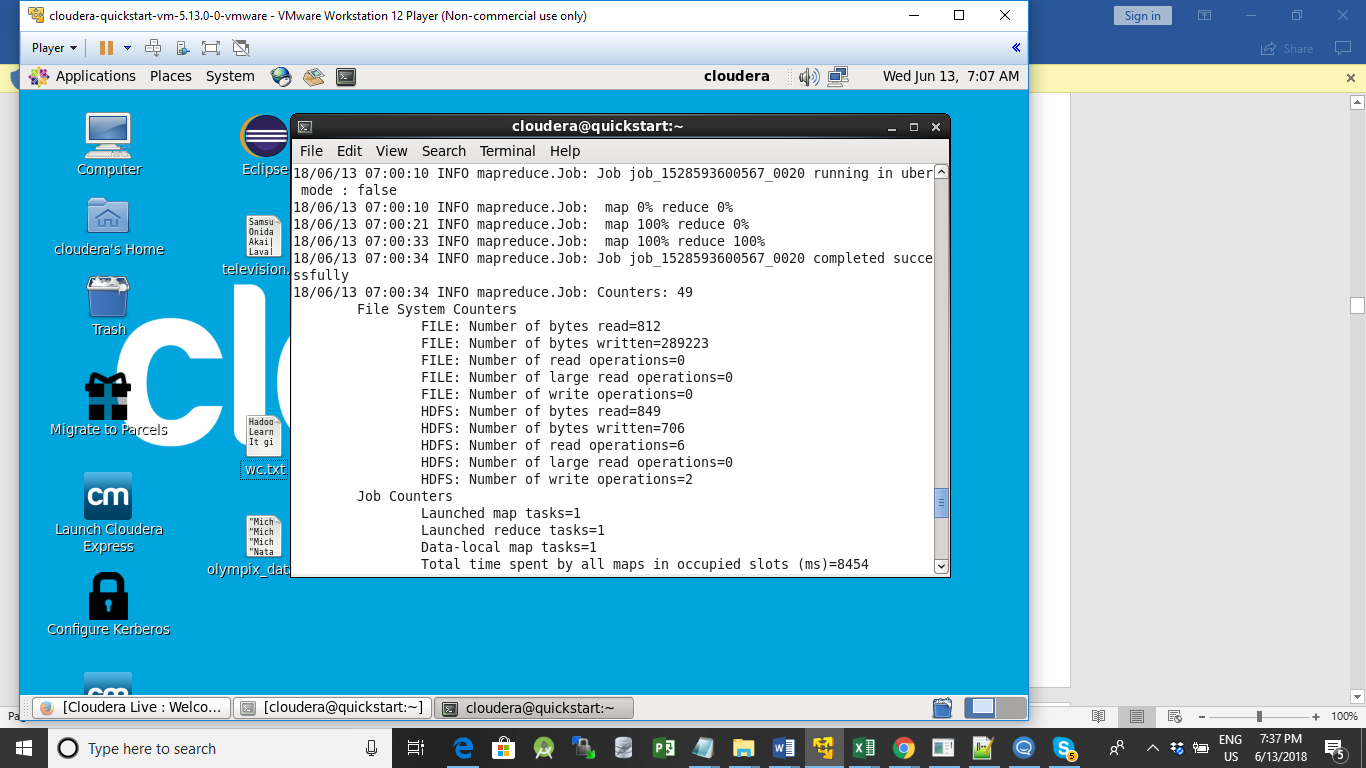
job.waitForCompletion(**true**);

}

}

**To run map reduce in Hadoop**

[cloudera@quickstart ~]$ hadoop jar task1.jar com.test.task2.Task1 /Ajay/television.txt /output2



To see the result Output

[cloudera@quickstart ~]$ hadoop fs -ls /output2

Found 2 items

-rw-r--r-- 1 cloudera supergroup 0 2018-06-10 12:26 /output2/\_SUCCESS

-rw-r--r-- 1 cloudera supergroup 742 2018-06-10 12:26 /output2/part-r-00000

[cloudera@quickstart ~]$ hadoop fs -cat /output2/part-r-000000

[cloudera@quickstart ~]$ hadoop fs -cat /output2/part-r-000000 Samsung|Optima|14|Madhya Pradesh|132401|14200

47 Onida|Lucid|18|Uttar Pradesh|232401|16200

90 Akai|Decent|16|Kerala|922401|12200

126 Lava|Attention|20|Assam|454601|24200

164 Zen|Super|14|Maharashtra|619082|9200

202 Samsung|Optima|14|Madhya Pradesh|132401|14200

249 Onida|Lucid|18|Uttar Pradesh|232401|16200

292 Onida|Decent|14|Uttar Pradesh|232401|16200

369 Lava|Attention|20|Assam|454601|24200

407 Zen|Super|14|Maharashtra|619082|9200

445 Samsung|Optima|14|Madhya Pradesh|132401|14200

532 Samsung|Decent|16|Kerala|922401|12200

571 Lava|Attention|20|Assam|454601|24200

609 Samsung|Super|14|Maharashtra|619082|9200

651 Samsung|Super|14|Maharashtra|619082|9200

693 Samsung|Super|14|Maharashtra|619082|9200

**As we can see in this all the enteries with “NA” are filtered**